Recherche linéaire et fusion de données par ajustement de faisceaux : application à la localisation par vision

Julien Michot

N° d'ordre, D.U. : 2085
EDSPIC : 505

UNIVERSITÉ BLAISE PASCAL - CLERMONT FERRAND II

ÉCOLE DOCTORALE
SCIENCES POUR L'INGENIEUR DE CLERMONT FERRAND

T H È S E

pour obtenir le grade de

Docteur d'Université

Spécialité : VISION POUR LA ROBOTIQUE

Présentée par

JULIEN MICHOT

Recherche linéaire et fusion de données par ajustement de faisceaux

Application à la localisation par vision

Thèse préparée au CEA Saclay,

soutenue publiquement le 9 décembre 2010

Jury :

Rapporteurs :	Pr. Ernest HIRSCH	LSIIT
	Pr. Éric MARCHAND	IRISA-INRIA
Examinateur :	Dr. Pierre GURDJOS	IRIT
Président :	Pr. Malik MALLEM	IBISC
Encadrants :	Pr. Adrien BARTOLI	ISIT
	Dr. François GASPARD	CEA-LIST
Directeur de thèse :	Pr. Jean-Marc LAVEST	IUT CLERMONT-FERRAND

“Dans les monts de la vérité, tu ne grimpes jamais en vain : ou bien tu arrives dès aujourd’hui à gagner de la hauteur, ou bien tu exerces tes forces pour pouvoir monter plus haut demain.”

Friedrich Nietzsche, Humain, trop humain II.

Remerciements

Je tiens tout d'abord à remercier Malik Mallem d'avoir accepté d'être le président de mon jury de thèse, ainsi qu'Ernest Hirsch, Éric Marchand et Pierre Gurdjos pour avoir eu la gentillesse d'évaluer mes travaux de recherche.

Je remercie également mes responsables de thèse : Jean-Marc Lavest, Adrien Bartoli et François Gaspard, pour avoir partagé leur formidable connaissance du domaine et de m'avoir fait découvrir une thématique de recherche particulièrement intéressante.

Mes remerciements vont aussi vers mes anciens collègues de travail : stagiaires, doctorants, postdocs, ingénieurs et chercheurs du laboratoire LVIC, sans oublier les secrétaires Annie et Élodie, pour leur sympathie et pour avoir formé un environnement de travail très agréable. Je remercie particulièrement les docteurs Pierre Lothe et Alexandre Eudes avec qui j'ai pu avoir de stimulants échanges scientifiques et techniques.

Merci aussi de m'avoir supporté durant la rédaction de ce mémoire ! ! !

Je remercie de plus les partenaires du projet Gyroviz, et notamment la société Sofresud qui a financé cette thèse, pour m'avoir donné l'opportunité d'expérimenter mes travaux de recherche en me fournissant diverses séquences de données avec une vérité terrain (qu'il n'est souvent pas aisé d'obtenir).

Je suis également très reconnaissant envers mes parents Brigitte et Pascal ainsi que mes deux sœurs Céline et Émilie qui m'ont toujours soutenu durant mes études et qui m'ont donné le goût des sciences en aiguisant ma curiosité et enseigné le sens des valeurs humaines.

Je remercie enfin ma chère et tendre, Minh Nguyet, pour sa formidable joie de vivre communicative et pour avoir magnifiquement égaillé cette fin de thèse.

Résumé

Les travaux présentés dans ce manuscrit concernent le domaine de la localisation et la reconstruction 3D par vision artificielle. Dans ce contexte, la trajectoire d'une caméra et la structure 3D de la scène filmée sont initialement estimées par des algorithmes linéaires puis optimisées par un algorithme non-linéaire, l'ajustement de faisceaux.

Cette thèse présente tout d'abord une technique de recherche de l'amplitude de déplacement (recherche linéaire), ou *line search* pour les algorithmes de minimisation itérative. La technique proposée est non itérative et peut être rapidement implantée dans un ajustement de faisceaux traditionnel. Cette technique appelée recherche linéaire algébrique globale (*G-ALS*), ainsi que sa variante à deux dimensions (*Two way-ALS*), accélèrent la convergence de l'algorithme d'ajustement de faisceaux. L'approximation de l'erreur de reprojection par une distance algébrique rend possible le calcul analytique d'une amplitude de déplacement efficace (ou de deux pour la variante *Two way-ALS*), par la résolution d'un polynôme de degré 3 (*G-ALS*) ou 5 (*Two way-ALS*). Nos expérimentations sur des données simulées et réelles montrent que cette amplitude, optimale en distance algébrique, est performante en distance euclidienne, et permet de réduire le temps de convergence des minimisations.

Une difficulté des algorithmes de localisation en temps réel par la vision (SLAM monoculaire) est que la trajectoire estimée est souvent affectée par des dérives : dérives d'orientation, de position et d'échelle. Puisque ces algorithmes sont incrémentaux, les erreurs et approximations sont cumulées tout au long de la trajectoire, et une dérive se forme sur la localisation globale. De plus, un système de localisation par vision peut toujours être ébloui ou utilisé dans des conditions qui ne permettent plus temporairement de calculer la localisation du système. Pour résoudre ces problèmes, nous proposons d'utiliser un capteur supplémentaire mesurant les déplacements de la caméra. Le type de capteur utilisé varie suivant l'application ciblée (un odomètre pour la localisation d'un véhicule, une centrale inertielle légère ou un système de navigation à guidage inertiel pour localiser une personne). Notre approche consiste à intégrer ces informations complémentaires directement dans l'ajustement de faisceaux, en ajoutant un terme de contrainte pondéré dans la fonction de coût. Nous évaluons trois méthodes permettant de sélectionner dynamiquement le coefficient de pondération et montrons que ces méthodes peuvent être employées dans un SLAM multi-capteur temps réel, avec différents types de contrainte, sur l'orientation ou sur la norme du déplacement de la caméra. La méthode est applicable pour tout autre terme de moindres carrés. Les expérimentations menées sur des séquences vidéo réelles montrent que cette technique d'ajustement de faisceaux contraint réduit les dérives observées avec les algorithmes de vision classiques. Ils améliorent ainsi la précision de la localisation globale du système.

Mots clés : ajustement de faisceaux, vision monoculaire, localisation, reconstruction 3D, fusion multi-capteur.

Abstract

The works presented in this manuscript are in the field of computer vision, and tackle the problem of real-time vision based localization and 3D reconstruction. In this context, the trajectory of a camera and the 3D structure of the filmed scene are initially estimated by linear algorithms and then optimized by a nonlinear algorithm, bundle adjustment.

The thesis first presents a new technique of line search, dedicated to the nonlinear minimization algorithms used in Structure-from-Motion. The proposed technique is not iterative and can be quickly installed in traditional bundle adjustment frameworks. This technique, called Global Algebraic Line Search (*G-ALS*), and its two-dimensional variant (*Two way-ALS*), accelerate the convergence of the bundle adjustment algorithm. The approximation of the reprojection error by an algebraic distance enables the analytical calculation of an effective displacement amplitude (or two amplitudes for the *Two way-ALS* variant) by solving a degree 3 (*G-ALS*) or 5 (*Two way-ALS*) polynomial. Our experiments, conducted on simulated and real data, show that this amplitude, which is optimal for the algebraic distance, is also efficient for the Euclidean distance and reduces the convergence time of minimizations.

One difficulty of real-time tracking algorithms (monocular SLAM) is that the estimated trajectory is often affected by drifts : on the absolute orientation, position and scale. Since these algorithms are incremental, errors and approximations are accumulated throughout the trajectory and cause global drifts. In addition, a tracking vision system can always be dazzled or used under conditions which prevented temporarily to calculate the location of the system. To solve these problems, we propose to use an additional sensor measuring the displacement of the camera. The type of sensor used will vary depending on the targeted application (an odometer for a vehicle, a lightweight inertial navigation system for a person). We propose to integrate this additional information directly into an extended bundle adjustment, by adding a constraint term in the weighted cost function. We evaluate three methods (based on machine learning or regularization) that dynamically select the weight associated to the constraint and show that these methods can be used in a real time multi-sensor SLAM, and validate them with different types of constraint on the orientation or on the scale. Experiments conducted on real video sequences show that this technique of constrained bundle adjustment reduces the drifts observed with the classical vision algorithms and improves the global accuracy of the positioning system.

Key words : bundle adjustment, monocular vision, localization, 3D reconstruction, multi-sensor data fusion.

Sommaire

Remerciements **3**

Résumé et abstract **5**

Introduction **11**

1 Éléments de base **17**
1.1 Conventions de notation . . . 18
1.2 Système et environnement . . . 20
1.3 Modélisation des capteurs . . . 22
1.4 Mesures de distance . . . 32

2 Estimation de paramètres et sélection de modèle **35**
2.1 Formalisation du problème . . . 36
2.2 Méthodes de moindres carrés . . . 38
2.3 Estimation robuste . . . 54
2.4 Sélection de complexité . . . 56
2.5 Conclusion . . . 61

3 Localisation et reconstruction 3D par vision **63**
3.1 Algorithmes de base . . . 64
3.2 Ajustement de faisceaux . . . 77
3.3 *Structure-from-Motion* . . . 86
3.4 Conclusion . . . 93

4 Recherche linéaire pour la reconstruction 3D **95**
4.1 État de l'art . . . 96
4.2 Recherche linéaire algébrique générique . . . 101
4.3 Recherche linéaire algébrique pour l'ajustement de faisceaux . . . 108
4.4 Résultats expérimentaux . . . 117
4.5 Discussion et conclusion . . . 133

5 Fusion de données par ajustement de faisceaux **135**
5.1 État de l'art . . . 136
5.2 Fusion de données d'incertitude inconnue . . . 142
5.3 Application au SLAM multi-capteur . . . 161
5.4 Résultats expérimentaux . . . 168
5.5 Discussion et conclusion . . . 183

Conclusion **185**

Bibliographie **187**

Table des matières **205**

Liste des figures **209**

Liste des tableaux **212**

Liste des algorithmes **213**

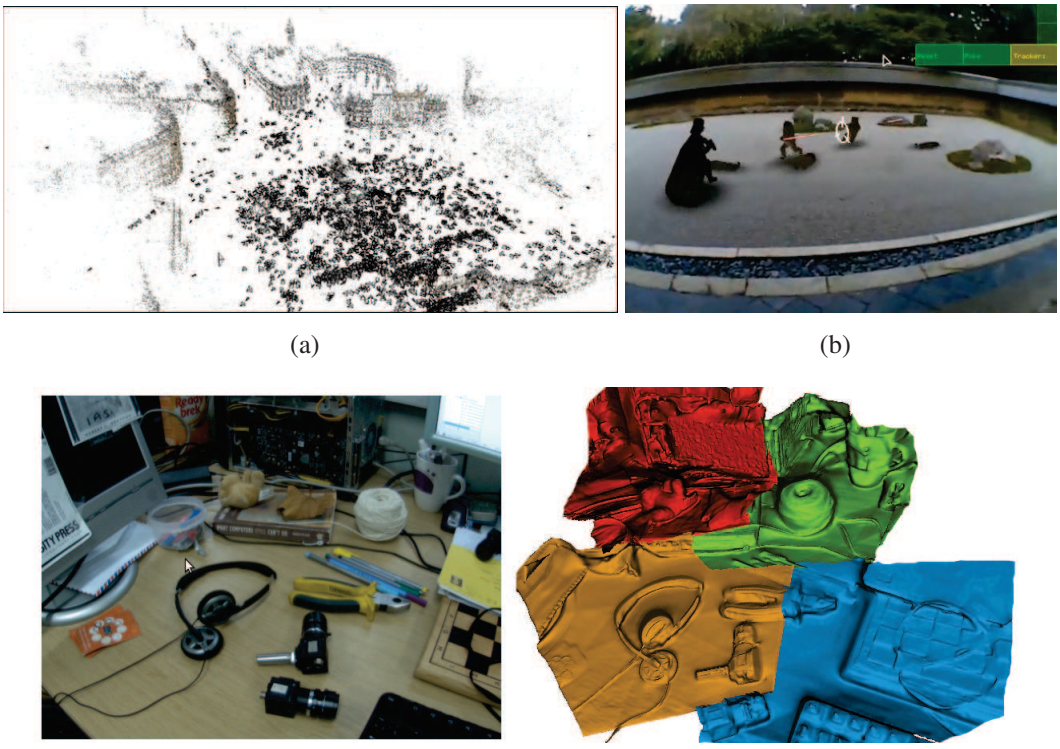
Introduction

La vision par ordinateur est un domaine de recherche avec un large champ d'applications. Les algorithmes de *structure à partir du mouvement* (*Structure-from-Motion*) par exemple, permettent aujourd'hui avec un simple appareil photographique ou une caméra vidéo, de reconstruire en trois dimensions un objet, une scène, un bâtiment voire une ville entière [Agarwal 2009, Frahm 2010, Strecha 2010]. Les applications possibles sont nombreuses : architecture et urbanisme, métrologie industrielle et rétro-ingénierie, cartographie pour la construction de données géographiques (SIG [1]), jeux vidéo localisés dans des scènes réelles, visites virtuelles, etc. Mais ces techniques permettent aussi de localiser un système dans un milieu connu ou inconnu, avec pour finalité, de multiples applications : le cinéma (post-production pour les effets spéciaux avec les logiciels `Boujou` de Vicon ou `MatchMover` d'Autodesk) ou la réalité augmentée [Klein 2007, Newcombe 2010] (localisation d'un *smartphone*) qui sur-impriment des objets ou des avatars virtuels dans les images. Il y a aussi la localisation et la navigation autonome de drones, de robots, de sous-marins ou de véhicules dans des milieux variés. Quelques unes de ces applications sont illustrées en figure 1. Toutes ces applications sont aujourd'hui possibles grâce aux importantes avancées réalisées depuis quelques dizaines d'années dans le domaine du *Structure-from-Motion*, et notamment la réalisation d'algorithmes temps réel et leur portage sur des systèmes embarqués.

Les algorithmes de *Structure-from-Motion* sont capables d'estimer les déplacements d'une caméra ou d'un appareil photographique et de reconstruire la structure de la scène seulement à partir des images 2D. Le procédé est le suivant : des éléments (ou primitives, par exemple des points 2D) de la scène filmée sont détectés puis suivis à travers l'ensemble des images. Le processus de reconstruction à partir du mouvement est ensuite composé de deux grandes étapes. La première étape consiste à estimer algébriquement les variables du problème, c'est-à-dire la trajectoire d'une caméra et la structure de la scène (les points 3D par exemple), puis de raffiner cette estimation par une méthode statistiquement optimale. Cette dernière étape est essentielle car c'est elle qui apporte la précision à la reconstruction de la scène. Cette étape est l'ajustement de faisceaux (*bundle adjustment*) [Slama 1980, Triggs 2000]. L'ajustement de faisceaux est le processus qui consiste à optimiser simultanément la trajectoire de la caméra et la structure de la scène. L'objectif est de minimiser les distances entre les éléments de la scène réelle détectés dans les images et les éléments obtenus lorsque la scène estimée est reprojetée dans les mêmes images.

Les travaux présentés dans cette thèse concernent l'ajustement de faisceaux et son application pour la reconstruction 3D hors ligne, et la localisation 3D en temps réel.

1. Système d'Information Géographique



(a) (b)

(c)

FIGURE 1 – Exemples d'applications des algorithmes de *Structure-from-Motion* : (a) une reconstruction de la place Trafalgar à londres [Agarwal 2009], (b) une application de réalité augmentée [Klein 2007], (c) une reconstruction dense en ligne d'un bureau avec une caméra à faible résolution [Newcombe 2010].

Problématiques et objectifs de la thèse

Les objectifs de la thèse sont doubles : accélérer la convergence de l'ajustement de faisceaux, et améliorer la qualité de la localisation en SLAM (*Simultaneous Localization And Mapping*) monoculaire par ajustement de faisceaux en intégrant des informations issues d'un second capteur.

L'ajustement de faisceaux est un processus coûteux. Si l'ajustement de faisceaux est le composant algorithmique qui apporte la précision, c'est aussi l'un des composants les plus complexes, les plus coûteux en temps de calcul. Le premier objectif de cette thèse est de réduire le temps nécessaire à la réalisation de l'ajustement de faisceaux dans les applications de *batch Structure-from-Motion*, tout en conservant la qualité de la reconstruction de la scène et de la localisation de la caméra. Ceci afin de converger plus rapidement vers la solution du problème et économiser ainsi les ressources du système.

La localisation en temps réel par les algorithmes de vision dérive. Les techniques de localisation basées uniquement sur la vision sont affectées par un certain nombre de problèmes : impossibilité d'estimation à cause de mauvaises conditions visuelles (occultation, faible quantité de texture, faible luminosité, etc.), dérives (en position, en orientation ou en échelle pour le monoculaire) dues au bruit, à la géométrie de la scène, aux approximations des modèles ainsi qu'aux minima locaux de l'ajustement de faisceaux. Pour palier ces difficultés, il est courant de combiner des informations complémentaires provenant de différentes sources : de caméras supplémentaires (stéréoscopique voire multi-caméra), de capteurs de mouvement (centrale inertielle [You 2001, Huster 2003, Strelow 2004, Hol 2006, Oskiper 2007, Armesto 2007, Gemeiner 2007, Hol 2007, Servant 2010], odomètre [Chenavier 1992, Eudes 2010a], etc.) ou de position (système de navigation, GPS [Agrawal 2006b], etc.), de capteurs mesurant l'environnement (laser LiDAR [Newman 2006, Mirisola 2007, Naikal 2009], radar, sonar [Ribas 2006], etc.) ou par l'utilisation de données issues d'autres méthodes (reconnaissance de lieu [Eade 2008, Angeli 2008], contraintes géométriques sur la scène [Lothe 2009] ou sur le déplacement, etc.). Le second objectif de cette thèse concerne la réduction des dérives de position, d'orientation ou d'échelle, observées sur l'estimation de la trajectoire d'une caméra par les algorithmes de *Structure-from-Motion* séquentiels temps réel (SLAM monoculaire). Pour cela, il sera nécessaire d'utiliser des informations extérieures, issues d'un second capteur mesurant les déplacements absolus du système, tout en conservant la contrainte de localisation en temps réel.

Contexte de la thèse

Cette thèse a été effectuée entre novembre 2007 et novembre 2010 au Laboratoire Vision et Ingénierie des Contenus (LVIC) du CEA LIST, à Saclay et a été financée par le projet de recherche Gyroviz. L'ensemble des travaux a été réalisé en cotutelle avec le laboratoire Comsee du LASMEA, à Clermont-Ferrand.

Contributions de la thèse

Les travaux réalisés durant cette thèse ont donné lieu à 4 principales contributions scientifiques et techniques.

Une technique de recherche linéaire adaptée aux problèmes de *Structure-from-Motion*. Nous avons proposé et étudié une nouvelle technique de recherche linéaire non itérative qui accélère la minimisation dans les problèmes de *Structure-from-Motion*. Le principe de la méthode est de simplifier la fonction de coût du problème en utilisant une distance algébrique en lieu et place de la distance Euclidienne, afin de calculer une amplitude de déplacement efficacement à chaque itération de l'optimisation.

Une deuxième contribution concerne la technique de recherche linéaire algébrique bidimensionnelle. Cette technique est une variante de la méthode précédente, dans laquelle nous

estimons deux amplitudes de déplacement pour l'ensemble des paramètres de la trajectoire de la caméra et pour l'ensemble des paramètres de la scène reconstruite.

Nous présentons ces deux techniques dans le cadre général des problèmes de *Structure-from-Motion* dans le chapitre 4, et décrivons comment elles peuvent être employées dans les ajustements de faisceaux calibrés ou non calibrés. Ces travaux ont été publiés dans les conférences CORESA 2009 [Michot 2009c], ORASIS 2009 [Michot 2009b] et BMVC 2009 [Michot 2009a].

Un système de localisation multi-capteur temps-réel par ajustement de faisceaux contraint. La troisième contribution est une méthode pour intégrer des informations génériques dans un algorithme de *Structure-from-Motion* incrémental. Nous proposons une solution à ce problème en intégrant un terme de régularisation pondéré dans la fonction de coût de l'ajustement de faisceaux local. Nous proposons trois heuristiques permettant de sélectionner dynamiquement les coefficients de pondération. Cette méthode est générique puisque les contraintes ajoutées peuvent être de différentes natures (lissage, fusion multi-capteur, autocalibrage, etc.)

Enfin, nous avons réalisé un système de localisation temps réel multi-capteur en utilisant la précédente méthode de fusion générique. Ce système a été validé sur deux types d'applications : la localisation d'un véhicule, dans laquelle un odomètre est utilisé pour réduire la dérive du facteur d'échelle de la trajectoire estimée ; l'application de la localisation en mouvement libre où une caméra et une centrale inertielle (ou un système de navigation à guidage inertiel) sont placés sur un opérateur se déplaçant à l'intérieur d'un bâtiment.

Ces techniques de fusion générique et de SLAM multi-capteur sont décrites dans le chapitre 5. Elles ont été publiées à la conférence internationale 3DPVT 2010 [Michot 2010a] et brevetées [Michot 2010b] (brevet en cours de révision).

Résumés des chapitres

La thèse est organisée en 5 chapitres que nous résumons ci-après. Elle se termine par les perspectives des travaux présentés.

Chapitre 1. Le premier chapitre introduit les éléments de base et notations nécessaires à la compréhension du mémoire. Après avoir introduit le système et les capteurs de localisation utilisés (caméra, centrale inertielle, etc.) et leurs modèles mathématiques, nous présentons les modèles de caméra perspective et la géométrie qui leur est associée. Nous décrivons enfin différentes notions de distance.

Chapitre 2. Le second chapitre présente quelques outils mathématiques qui seront utilisés dans les chapitres suivants, et notamment les techniques de résolution des problèmes inverses pour lesquels il s'agit de retrouver les paramètres d'un modèle à partir d'observations. Plus particulièrement, nous présentons les techniques d'optimisation non-linéaire non contrainte adaptées

aux fonctions de coût de moindres carrés, ainsi que les méthodes d'optimisation robuste et multiobjectif. Enfin, nous abordons le problème de la sélection de complexité d'un modèle et des techniques associées.

Chapitre 3. Le troisième chapitre décrit des algorithmes de localisation et de reconstruction 3D par vision, et plus particulièrement les techniques de structure à partir du mouvement (*Structure-from-Motion*). Après avoir présenté les algorithmes de base en vision par ordinateur (détection, description et appariement de points d'intérêt, triangulation de point, calcul de pose), nous présentons les deux grandes catégories d'algorithmes de *Structure-from-Motion* : le *batch Structure-from-Motion*, où toutes les images sont traitées simultanément, et le *Structure-from-Motion* incrémental ou séquentiel (SLAM), dans lequel les images sont traitées les unes après les autres.

Chapitre 4. Le quatrième chapitre présente une méthode permettant d'accélérer la convergence des optimisations non-linéaires dans les méthodes de *Structure-from-Motion*. Après avoir présenté les différentes méthodes d'optimisation de l'état de l'art, nous présentons une nouvelle technique de recherche linéaire adaptée aux problèmes de *Structure-from-Motion*. Nous détaillons ensuite comment cette technique peut être employée dans un algorithme d'ajustement de faisceaux, dans le cadre d'une application de reconstruction 3D hors ligne par *batch Structure-from-Motion*.

Chapitre 5. Le cinquième et dernier chapitre décrit une méthode générique permettant d'intégrer des informations de localisation issues d'un capteur ou d'un algorithme extérieur dans un ajustement de faisceaux. Ces informations supplémentaires sont insérées sous la forme de contraintes de régularisation pondérées dans la fonction de coût de l'ajustement de faisceaux. Trois techniques permettant de sélectionner dynamiquement les coefficients de pondération associées aux contraintes sont étudiées. Enfin, nous proposons d'employer cette technique dans le cadre d'un algorithme de localisation temps réel (SLAM) multi-capteur dans lequel un second capteur (odomètre, centrale inertielle ou système de navigation à guidage inertiel) est utilisé afin de contraindre la trajectoire estimée par un SLAM monoculaire.

CHAPITRE 1

Éléments de base

Dans ce premier chapitre nous introduisons les éléments de base et notations nécessaires à la compréhension du mémoire. Après avoir introduit le système et les capteurs de localisation utilisés (caméra, centrale inertielle, etc.) et leurs modèles mathématiques, nous présentons les caméras perspectives et la géométrie qui leur est associée. Nous décrivons enfin la notion de distance et ses différentes représentations.

This first chapter introduces the basic elements and notation necessary for the understanding of this memory. After introducing the system and the sensors used (camera, inertial system, ...) and their mathematical models, we present the perspective cameras and their associated geometry. Finally, we describe the concept of distance and its various representations.

1.1 Conventions de notation

Dans ce mémoire, nous adoptons les notations suivantes :

Éléments algébriques

a	un scalaire *(caractère italique)*
$\mathbf{x}$	un vecteur *(caractère gras)* $\mathbf{x}^\top = [x_1, x_2, \ldots, x_p]$
$\mathcal{P}_{m\times n}$	une matrice de m lignes et n colonnes *(caractères calligraphiques)*, avec $\mathcal{P}_{m\times n} = \begin{pmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,n} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{m,1} & P_{m,2} & \cdots & P_{m,n} \end{pmatrix}$.
$\mathfrak{M}$	un espace, ou une variété *(caractères de Fraktur)*
$\mathcal{I}_n$	matrice identité de taille $n \times n$

Opérateurs mathématiques

$Card(\mathbf{x})$	cardinalité d'un ensemble ou taille d'un vecteur
$\|\mathbf{x}\|_n$	norme n du vecteur $\mathbf{x}$, $\|\mathbf{x}\|_n = (x_1 + \ldots x_p)^{\frac{1}{n}}$
$\|\mathbf{x}\|$	norme 2 du vecteur $\mathbf{x}$, $\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \mathbf{x}}$
$\|\mathbf{x}\|_\Sigma$	norme de Mahalanobis du vecteur $\mathbf{x}$, telle que $\|\mathbf{x}\|_\Sigma = \sqrt{\mathbf{x}^\top \Sigma \mathbf{x}}$
$^\top$	opérateur de transposition
$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^\top \mathbf{y}$	produit scalaire entre les vecteurs $\mathbf{x}$ et $\mathbf{y}$
$\mathbf{x} \times \mathbf{y}$	produit vectoriel entre les vecteurs $\mathbf{x}$ et $\mathbf{y}$
$\mathrm{diag}(\mathbf{x})$	fonction qui génère une matrice dont la diagonale est $\mathbf{x}$
$\mathrm{vect}(\mathcal{P})$	fonction qui accumule les éléments de la matrice $\mathcal{P}$ en un vecteur
$[\mathbf{x}]_\times$	la matrice antisymétrique de $\mathbf{x}$, telle que $[\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}$ - cette matrice permet de réaliser un produit vectoriel $\mathbf{x} \times \mathbf{y}$ sous la forme d'un produit matriciel puisque $[\mathbf{x}]_\times \mathbf{y} = \mathbf{x} \times \mathbf{y}$
$[\mathbf{xy}]$	opérateur de concaténation de deux vecteurs, en un troisième vecteur
$\Psi(\mathbf{q})$	opérateur permettant de passer des coordonnées homogènes aux coordonnées affines avec $\tilde{\mathbf{q}} = \Psi(\mathbf{q}) = \frac{1}{q_p} \begin{pmatrix} q_1 \\ \vdots \\ q_{p-1} \end{pmatrix}$

Repères et entités

$\mathcal{W}$	un repère local
$\mathcal{W}_g$	le repère terrestre (global) (WGS84 par exemple)
$\mathcal{W}_c$	le repère local d'une caméra
$\mathcal{W}_x$	le repère local de la centrale inertielle
$\mathcal{W}_n$	le repère local de la centrale de navigation
$\mathcal{W}_o$	le repère local de l'odomètre
${}^{\mathcal{W}_k}\mathbf{x}$	les paramètres $\mathbf{x}$ sont exprimés dans le repère $\mathcal{W}_k$

Éléments géométriques

$\mathbf{Q}_j$	les coordonnées homogènes d'un point 3D Q_j
$\mathbf{q}_j^i$	les coordonnées homogènes d'un point 2D, correspondant à la projection du point 3D Q_j sur l'image c_i
$\mathcal{P}_i$	la matrice de projection de la caméra c_i, de taille 3×4
$\mathcal{R}$	une matrice de rotation appartenant au groupe orthonormal $\mathcal{R} \in SO(3)$
$\mathbf{t}_i$	position de la caméra c_i
$\mathcal{T}_i$	pose de la caméra c_i, comprenant sa position et orientation

Paramètres de dimensionnalité

N_n	nombre de points 3D du modèle
N_m	nombre d'images localisées (poses de la caméra)
N_z	taille du vecteur d'observation $\hat{\mathbf{z}}$
N_p	taille du vecteur des paramètres du modèle $\mathbf{x}$
N_p^c	taille du vecteur des paramètres des caméras
N_p^Q	taille du vecteur des paramètres des points 3D
N_ν	nombre d'observations de confiance (*inliers*)
N_{ν_i}	nombre d'observations de confiance de la pose i

Paramètres pour l'optimisation

$\hat{\mathbf{z}}_t^k$	un vecteur d'observations du capteur k à l'instant t
σ_z	la variance associée à l'observation z
$\mathbf{x}$	le vecteur de paramètres du modèle, de taille N_p
$\mathfrak{M}$	l'espace des paramètres du modèle
$\mathbf{x}^{(k)}$	les paramètres du modèle estimés à l'itération k
$\mathbf{x}^*$	les paramètres du modèle optimal
$p(\mathbf{x})$	une probabilité
$\mathcal{J}_F(\mathbf{x}, \boldsymbol{\delta})$	la matrice Jacobienne de la fonction F sur les paramètres $\boldsymbol{\delta}$ évaluée en $\mathbf{x}$
$\mathcal{H}_F(\mathbf{x}, \boldsymbol{\delta})$	la matrice Hessienne de la fonction F sur les paramètres $\boldsymbol{\delta}$ évaluée en $\mathbf{x}$

1.2 Système et environnement

L'objectif de cette thèse est de localiser un système dans un environnement inconnu, et de reconstruire l'environnement. Nous définissons le système comme un élément mobile (une personne, une voiture, etc.) muni de capteurs mesurant le monde ou les déplacements du système dans l'environnement. Notons $\mathbf{x}_t^{\mathcal{P}}$ le vecteur constitué des informations de localisation (de position $\mathbf{t}_t \in \mathbb{R}^3$ et d'orientation $\mathcal{R}_t \in SO(3)$) du système. Puisque que le système est mobile, sa localisation évolue au cours du temps, indicé par la variable t. À l'inverse, l'environnement, composé d'éléments inconnus mais fixes notés $\mathbf{x}^{\mathcal{Q}}$, sera dépourvu d'indice temporel.

Notre approche consiste à localiser un système en reconstruisant simultanément l'environnement à partir des mesures de divers capteurs. Pour simplifier les notations du problème, nous considérerons que le repère du système est le repère de la caméra, c'est-à-dire que le centre du système complet en mouvement est le centre optique de la caméra, et l'on cherchera à localiser celui-ci. Puisque la caméra est rigidement liée au système, la position du système pourra être obtenue par une simple transformation Euclidienne a priori connue (par l'étalonnage des changements de repère $\mathcal{C}_{s/k}$ entre les différents composants du système).

Les paramètres de notre modèle se composent de la localisation du système $\mathbf{x}_t^{\mathcal{P}}$ (ou d'un historique $\mathbf{x}_{t\in[t_1,t_2]}^{\mathcal{P}}$ sur la période $[t_1, t_2]$) et de l'environnement $\mathbf{x}^{\mathcal{Q}}$. Nous notons $\mathfrak{M}$ l'espace des vecteurs de paramètres valides ($\mathbf{x} \in \mathfrak{M}$).

1.2.1 Modélisation de la dynamique

La modélisation de la dynamique d'un système mobile permet de prédire la position et/ou l'orientation qu'aura le système à un instant futur donné. La dynamique du système dépend du contexte applicatif et peut être modélisée de façon précise si l'on possède des connaissances sur les mouvements du système : degrés de liberté, amplitudes des variations, commandes envoyées aux actionneurs, etc. Dans notre thèse nous avons considéré plusieurs contextes applicatifs : un piéton circulant de manière aléatoire et une voiture circulant dans un milieu urbain.

Dans le cas de l'application de localisation d'une personne, nous avons très peu d'informations extéroceptives permettant de prédire le mouvement à l'avance puisque l'utilisateur va se déplacer comme il le souhaite. Nous n'avons donc aucun a priori sur la trajectoire de la personne et il nous faut dès lors trouver un modèle de prédiction générique qui corresponde le plus à la réalité en n'ayant comme seule information que les mouvements précédents du système. Dans la littérature [Zhang 1992], les modèles classiques de mouvement basés uniquement sur le passé sont les suivants :

- modèle de position constante
- modèle de vitesse constante en translation et en rotation
- modèle d'accélération constante en translation, vitesse constante en rotation
- modèle de translation générale, vitesse constante en rotation

Le modèle le plus approprié pour nos applications est le modèle à vitesse constante en translation et en rotation. Avec ce modèle, nous définissons la fonction de prédiction de l'état du système X, qui prédit les paramètres du système $\mathbf{x} : \mathfrak{M} \rightarrow \mathfrak{M}$ à l'instant $t + dt$ à partir de l'état

précédent $\mathbf{x}_t$, telle que

$$\mathbf{x}_{t+dt} = X_{dt}(\mathbf{x}_t). \tag{1.1}$$

Dans le cadre du modèle à vitesse constante en translation et en rotation, la fonction de prédiction du modèle est définie par X_{dt} :

$$\mathcal{R}_{t+dt} = \mathcal{R}_t \, \mathrm{rot_e}\left(\boldsymbol{\omega}_t dt\right) \tag{1.2}$$

$$\mathbf{t}_{t+dt} = \mathbf{t}_t + \mathbf{v}_t dt, \tag{1.3}$$

où $\mathbf{v}_t$ est la vitesse en translation du système à l'instant t et $\boldsymbol{\omega}_t$ sa vitesse de rotation (ou vitesse angulaire, en angles d'Euler). $\mathrm{rot_e}\left(\mathbf{v}\right)$ est la fonction de transformation des angles en une matrice de rotation (voir le paragraphe sur les paramètres extrinsèques 1.3.1.2). Nous verrons que suivant les capteurs mis en jeu dans l'application, ces valeurs de vitesse pourront soit être directement les mesures des capteurs, soit estimées à l'aide de l'historique de localisation du système $\mathbf{x}^{\mathcal{P}}_{t\in[t_1,t_2]}$.

La fonction de prédiction ne permet pas de localiser un système de manière précise sur le long terme et il faut pour cela utiliser de nouvelles observations informant de l'état du système ou de l'environnement. Ces informations sont issues des capteurs placés sur le système.

1.2.2 Repères du système

Nous donnons un aperçu du système physique dans la figure 1.1. Le système mobile possède son propre repère noté $\mathcal{W}_s$ et évolue dans l'environnement, symbolisé par le repère monde $\mathcal{W}_g$. Le système possède plusieurs capteurs (caméra, odomètre, ...) qui lui sont rigidement liés et

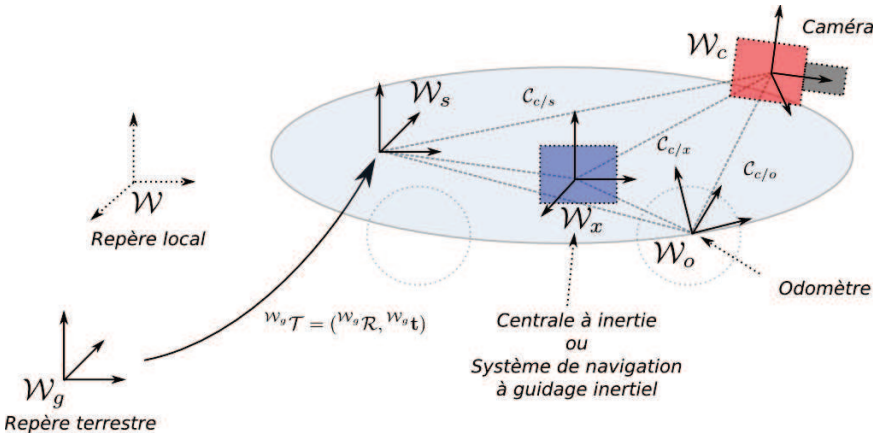


FIGURE 1.1 – Aperçu du système mobile et des différents repères.

qui délivrent des mesures dans leur propre repère local ($\mathcal{W}_c$, $\mathcal{W}_o$, ...). Pour simplifier les notations du problème, nous prenons l'hypothèse que les repères système et caméra sont confondus $\mathcal{W}_s = \mathcal{W}_c$ (donc $\mathcal{C}_{c/s} = \mathcal{I}$). $\mathcal{W}$ est un repère local quelconque dans lequel nous localiserons le système lorsque nous n'aurons pas d'information de localisation absolue (provenant d'un GPS par exemple).

1.3 Modélisation des capteurs

Un capteur est un dispositif qui fournit une mesure (ou observation) bruitée sur le monde extérieur (environnement). Le processus de mesure à l'instant t, réalisé par le capteur k, peut être modélisé par une fonction de mesure $h : \mathbb{R}^{N_p} \rightarrow \mathbb{R}^{N_z}$ qui, à partir des paramètres $\mathbf{x}$ du modèle que l'on a de l'environnement et du déplacement du système, délivre ses observations $\hat{\mathbf{z}}_t^k$ telles que

$$^{\mathcal{W}_k}\hat{\mathbf{z}}_t^k = h_t^k(\mathbf{x}_t) + \mathbf{e}_t^k, \tag{1.4}$$

où $^{\mathcal{W}_k}\hat{\mathbf{z}}_t^k$ est le vecteur d'observation de taille N_z, délivré par le capteur k à l'instant t et exprimé dans le repère $\mathcal{W}_k$. $\mathbf{e}_t^k$ modélise le bruit du processus de projection de $\mathbb{R}^{N_p}$ dans $\mathbb{R}^{N_z}$. Il peut être parfois décomposé en un vecteur de biais $\mathbf{b}_t^k$ et un vecteur de bruit Gaussien blanc (centré) $\mathbf{n}_t^k$, tel que $\mathbf{e}_t^k = \mathbf{b}_t^k + \mathbf{n}_t^k$. Le vecteur $\mathbf{x}_t$ représentera tout au long de ce mémoire, l'image ou le modèle que l'on a du monde, à l'instant t, ainsi que l'historique de la trajectoire du système.

Dans cette thèse nous utilisons plusieurs capteurs, délivrant directement (IMU/INS, odomètre, GPS) ou indirectement (caméra) une information sur la localisation du système.

1.3.1 Caméra sténopé

Une caméra est un capteur fournissant de nombreuses informations sur le monde extérieur. À l'instant t, une caméra capture une image et enregistre ainsi une grande quantité d'informations photométriques. Ces images peuvent être traitées afin d'obtenir des informations de plus haut niveau : points d'intérêt, contours, ... [Forsyth 2002]. Dans cette thèse nous nous intéressons principalement à la reconstruction de points. Par conséquent, nous nous attacherons à détecter dans les images des points 2D, projection de points 3D sur le plan image, appelés points d'intérêt. Le processus de détection est détaillé dans la section 3.1.1.1. Le vecteur d'observation $^{\mathcal{W}_c}\hat{\mathbf{z}}_t^c$ sera constitué des deux coordonnées $\tilde{\hat{\mathbf{q}}}^\top = [\hat{q}_1 \hat{q}_2]$ exprimées sur le plan image des points d'intérêt 2D (voir figure 1.2). Ce vecteur sera donc de taille $2N_z^c$, où N_z^c est le nombre de points d'intérêt détectés : $^{\mathcal{W}_c}\hat{\mathbf{z}}_t^c = [\ldots \tilde{\hat{\mathbf{q}}} \ldots]$

La modélisation d'une caméra définit la manière dont un élément 3D se projette sur le plan image de la caméra. Il existe plusieurs formalisations du processus de projection dans l'image, caractérisant les différents modèles de caméra utilisés (camera sténopé, *fisheye*, omnidirectionnelle, etc.). Un modèle courant est le modèle sténopé, représenté sur la figure 1.2.

Dans le modèle sténopé, la caméra est représentée par un plan 2D appelé le plan image $\boldsymbol{\pi}_{img}$ et un point 3D, le centre optique (ou de projection) que l'on notera $\mathbf{t}$. La distance entre ce centre optique et le plan image est la focale f de la caméra. Elle est exprimée en unité métrique. La projection d'un point 3D sur le plan image est opérée en dessinant un rayon optique partant du point 3D et allant vers le centre optique de la caméra. L'intersection formée par ce rayon optique et le plan image correspond à l'image du point 3D. Pour exprimer mathématiquement cette projection, nous introduisons le repère local de la caméra $\mathcal{W}_c$. Son origine, le centre optique $\mathbf{t}$, coïncide avec l'axe z, aussi appelé axe optique (ou axe principal) et est orthogonal au plan image. Cet axe principal rencontre le plan image au point principal $\mathbf{c}$.

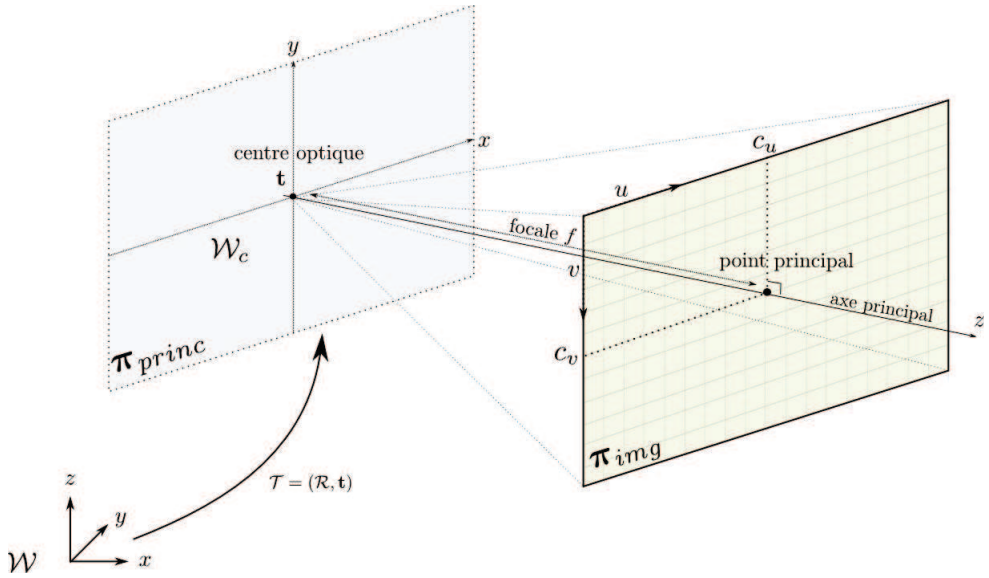


FIGURE 1.2 – Le modèle géométrique d'une caméra sténopé.

1.3.1.1 Caméra projective

Avant de définir la caméra projective finie, nous présentons un cadre théorique, la géométrie projective.

Géométrie projective. On définit l'espace projectif de dimension n, noté $\mathbb{P}^n$, comme l'ensemble des classes d'équivalence de $\mathbb{R}^{n+1}$ formé pour la relation $\sim$ telle que :

$$\mathbf{p} \sim \mathbf{q} \iff \exists \lambda \in \mathbb{R}^* | \mathbf{p} = \lambda \mathbf{q}. \tag{1.5}$$

Si des études théoriques de ces espaces existent, nous nous intéresserons dans nos travaux à la géométrie projective qui leur est associée et qui permet en particulier de formaliser la notion de point à l'infini dans les espaces affines.

Un vecteur de l'espace projectif $\mathbb{P}^n$ aura pour coordonnées :

$$\mathbf{q} = [q_1 \dots q_{n+1}]^\top, \tag{1.6}$$

avec les q_i non tous nuls. Si le dernier paramètre q_{n+1} est non nul, alors $\mathbf{q}$ représente le vecteur $\tilde{\mathbf{q}}$ de $\mathbb{R}^n$ avec $\tilde{\mathbf{q}} = [q_1/q_{n+1} \dots q_n/q_{n+1}]^\top$. Dans le cas contraire, le vecteur $\mathbf{q}$ décrit un point à l'infini. Les coordonnées $\mathbf{q}$ sont appelées *coordonnées homogènes* de $\tilde{\mathbf{q}}$. Dans l'ensemble du mémoire, l'utilisation du *tilde* indiquera que les coordonnées utilisées sont non-homogènes ou affines. La fonction qui permet de passer des coordonnées homogènes aux coordonnées affines est notée $\Psi : \mathbb{P}^n \to \mathbb{R}^n$ et est définie par $\Psi(\mathbf{q}) = \tilde{\mathbf{q}} = [q_1/q_{n+1} \dots q_n/q_{n+1}]^\top$ (voir la section 1.1).

Caméra projective. La caméra projective est la représentation le plus générique du modèle sténopé. Elle définit la projection d'un point 3D de coordonnées homogènes $\mathbf{Q}_j$ par l'équation

suivante :

$$^{\mathcal{W}_c}\mathbf{q}_j^i = \mathcal{P}_i \mathcal{W} \mathbf{Q}_j, \tag{1.7}$$

où $\mathbf{q}_j^i$ est le vecteur des coordonnées homogènes du point 2D exprimées dans le plan image de la caméra c_i. On appelle *matrice de projection* la matrice de taille 3×4 que l'on notera $\mathcal{P}$. Cette matrice est de forme générale dans le cas d'une caméra projective :

$$\mathcal{P} = \begin{pmatrix} p_{1,1} & p_{1,2} & p_{1,3} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,2} & p_{2,4} \\ p_{3,1} & p_{3,2} & p_{3,2} & p_{3,4} \end{pmatrix}, \tag{1.8}$$

avec $p_{i,j} \in \mathbb{R}$.

La matrice de projection d'une caméra projective possède 11 degrés de liberté (12 moins un facteur multiplicatif). Ce modèle permet aussi de représenter les caméras dont le centre de projection se situe à l'infini (les caméras affines par exemple). La fonction d'observation de la caméra est définie en concaténant les prédictions, en coordonnées non-homogènes :

$$h_{t_i}^c(\mathbf{x}) = [\ldots h_{i,j}^c(\mathbf{x}) \ldots], \quad \text{avec} \quad h_{i,j}^c(\mathbf{x}) = \Psi(\mathcal{P}_i \mathbf{Q}_j), \tag{1.9}$$

et où t_i est le temps de l'image i. Par la suite, on utilisera l'indice i en lieu et place de l'indice temporel t_i afin d'alléger les notations.

1.3.1.2 Caméra perspective

Dans le modèle de caméra perspective, la matrice de projection peut être décomposée en deux parties : le changement de repère $\mathcal{T}$ (du repère global $\mathcal{W}_g$ vers le repère local de la caméra $\mathcal{W}_c$) , et la projection $\mathcal{K}$ en elle même (du repère 3D local de la caméra au repère image 2D). Nous définissons la décomposition suivante :

$$\mathcal{P}_{3\times4} = \mathcal{K}_{3\times3} \mathcal{T}_{3\times4}. \tag{1.10}$$

La matrice de changement de repère $\mathcal{T}$ contient les paramètres extrinsèques de la caméra et la matrice $\mathcal{K}$, les paramètres intrinsèques.

Paramètres intrinsèques. Les coordonnées d'une image numérique sont définies dans le repère image (u, v), exprimé en pixels, et dont l'origine se situe en haut à gauche de l'image. Pour décrire la projection d'un point dans ce repère image, nous introduisons les paramètres suivants : nous considérons que les pixels du capteur CCD de la caméra ont une taille $d_u \times d_v$, exprimés en pixels. Les deux composantes u et v de la focale sont définies par $f_u = f/d_u$ et $f_v = f/d_v$ et sont exprimées en pixels. Le vecteur $\mathbf{c}^\top = [c_u c_v]$ représente le point principal de la caméra. s est le facteur de biais et modélise la non-orthogonalité des axes u et v du capteur. L'ensemble de ces paramètres compose la matrice de calibrage $\mathcal{K}$ de la caméra. Cette matrice,

affine, décrit la projection d'un point 3D exprimé dans le repère local de la caméra $\mathcal{W}_c$ dans le repère de l'image (en pixels). Nous définissons la matrice de calibrage $\mathcal{K}$ par

$$\mathcal{K} = \begin{pmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix}. \tag{1.11}$$

Les paramètres intrinsèques d'une caméra varient suivant les modèles de caméra et les modèles d'objectif utilisés. Ces paramètres varient même entre deux caméras ou deux objectifs de même modèle et peuvent évoluer au cours du temps (chocs, manipulations diverses, variation de la température extérieure, etc.). Il existe donc beaucoup de facteurs qui font que la matrice de calibrage évolue au cours du temps. Pour connaître les valeurs de ces paramètres, il est nécessaire de procéder à un étalonnage (ou calibrage, *calibration* en anglais) de la caméra (voir la section 1.3.1.4).

Paramètres extrinsèques. Les points 3D ne sont généralement pas exprimés dans le repère local de la caméra, mais dans un repère fixe, le repère global (ou objet) $\mathcal{W}_g$. Le transfert entre ces deux repères est donné par une transformation rigide $\mathcal{T}$, composée d'une rotation $\mathcal{R}$ et d'une translation $\mathbf{t}$, telle que $\mathcal{T} = \begin{pmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}$. Ce changement de repère constitue les paramètres extrinsèques ou la pose de la caméra. La matrice de rotation $\mathcal{R}^\top$ peut être interprétée comme l'orientation de la caméra et la translation $-\mathcal{R}^\top\mathbf{t}$ comme la position du centre optique de la caméra dans le repère global $\mathcal{W}_g$.

L'orientation de la caméra est donnée par le matrice de rotation $\mathcal{R}^\top$, cependant celle-ci possède bien plus de paramètres (9) qu'il n'en faut pour paramétrer une rotation en 3D (3). Plusieurs alternatives existent pour résoudre ce problème de paramétrisation des angles. Dans cette thèse, nous utilisons deux différentes représentations : la matrice de rotation $\mathcal{R}$, les angles d'Euler $\mathbf{e} = (\alpha, \beta, \gamma)$. Pour changer la paramétrisation de l'orientation nous définissons la fonction (et la fonction inverse) suivante :

– $\text{rot}_\text{e}(\mathbf{e})$ transforme un vecteur d'angles d'Euler en une matrice de rotation $\mathcal{R}$.

Dans notre application nous souhaitons estimer la trajectoire d'une caméra en utilisant plusieurs images. Pour faire simple, nous désignerons sous le terme *pose de caméra* la position et l'orientation d'une caméra pour une image donnée. Avec le modèle de caméra perspective, nous considérons que les paramètres intrinsèques des caméras sont connus, ainsi, le vecteur de paramètres du modèle $\mathbf{x}_\mathcal{P}$ représentant les caméras n'est donc constitué que des paramètres de pose (extrinsèques) de toutes les caméras : $\mathbf{x}_\mathcal{P} = [\ldots \mathbf{P}_i \ldots]$, avec $\mathbf{P}_i = [\text{vect}(\mathcal{R}_i)\tilde{\mathbf{t}}_i]$, la pose d'une caméra.

1.3.1.3 Modélisation de la distorsion optique

Le modèle sténopé seul ne suffit pas toujours à modéliser correctement les caméras réelles. En effet les objectifs des caméras introduisent une distorsion de la projection des points sur

l'image. La distorsion se manifeste selon une composante radiale et une composante tangentielle. Cette dernière étant beaucoup moins prononcée, nous avons considéré uniquement dans notre modèle la distorsion radiale, qui suffit généralement à approximer les distorsions optiques [Mikhail 2001].

Modéliser la distorsion revient à déterminer la fonction qui au point $\tilde{\mathbf{q}}$ associe le point corrigé en distorsion $\tilde{\mathbf{q}}_{sd}$. La distorsion radiale peut être paramétrée par un polynôme dont le degré peut aller jusqu'à 5 avec des objectifs grand-angle. Les coefficients du polynôme de distorsion radiale sont notés $(a_1, \ldots, a_i)$ où $i \in \mathbb{N}$ est le degré du polynôme (généralement de 1 à 5). Notons r la distance normalisée au carré entre le point $\tilde{\mathbf{q}}$ et le point principal $\mathbf{c}$, telle que $r = \left(\frac{\tilde{\mathbf{q}}_1 - c_u}{f_u}\right)^2 + \left(\frac{\tilde{\mathbf{q}}_2 - c_v}{f_v}\right)^2$. Le point corrigé en distorsion radiale est

$$\tilde{\mathbf{q}}_{sd} = \tilde{\mathbf{q}} + (\tilde{\mathbf{q}} - \mathbf{c}) \sum_i a_i r^i. \tag{1.12}$$

Dans la pratique nous avons modélisé les paramètres de distorsions radiale par 5 coefficients. Ces paramètres de distorsion peuvent être estimés lors d'une procédure d'étalonnage.

1.3.1.4 Étalonnage de caméra

L'étalonnage de caméra consiste à déterminer de manière précise les paramètres internes (ou intrinsèques) de la caméra. Ces paramètres internes se composent des deux composantes de la focale fixe f_u, f_v et du point principal $\mathbf{c}$. Nous supposons que les pixels sont carrés ($d_u = d_v$), et le facteur de biais est donc nul ($s = 0$). À ces 4 paramètres s'ajoutent les 5 paramètres de distorsions radiale. Il y a donc en tout 9 paramètres à estimer pour chaque caméra. En pratique, les paramètres extrinsèques, c'est-à-dire la pose comprenant l'orientation et la position de la caméra, sont estimés simultanément mais puisque dans notre application la caméra est mobile, ces valeurs ne sont pas utilisées.

Pour réaliser cette opération, il existe différentes procédés [Lavest 1998, Peter Sturm 1999, Zhang 2000, Gurdjos 2003] et deux principales bibliothèques libres [Bradski 2000, Bouguet 2008]. Le procédé le plus simple est de prendre plusieurs images d'une mire de calibrage (une mire plane avec des pastilles réfléchissantes dont les position 2D sont connues) et de chercher les paramètres de la caméra qui minimise l'erreur de reprojection (distance entre les observations et les positions théoriques, voir la section 3.2) de chaque point de la mire. Notons aussi qu'il est possible d'auto-étalonner une caméra [Pollefeys 1997] sans utiliser de mire de calibrage à l'aide de la conique absolue, sous certaines conditions (pas de mouvement critiques notamment [Sturm 1997]), voir la section 3.3.1.2 page 87.

Dans la problématique de reconstruction d'un environnement urbain à partir d'un grand nombre d'images issues d'Internet, certain auteurs [Snavely 2008a] montrent que l'on peut faire certaines hypothèses : le point principal de la caméra se situe au milieu de l'image, un seul paramètre de focale et deux paramètres de distorsion radiale suffisent pour reconstruire de manière efficace une ville. Ils proposent dans une première étape de trouver la focale du modèle de l'appareil photo à partir soit des données *Exif* des images, soit à partir d'une base de données de mod-

èles/focales. Et d'autre part, d'optimiser ces trois derniers paramètres (focale et deux paramètres de distorsion radiale) lors d'un ajustement de faisceaux (avec le logiciel *Bundler*[1]).

Dans les travaux présentés dans ce rapport nous considérons que les caméras sont calibrées en distorsion radiale (ou les images sont rectifiées). Afin d'alléger les notations, nous considérons que les observations 2D $\mathbf{q}$ données par le capteur caméra sont corrigées en distorsion radiales. Dans la première partie de ce mémoire, les paramètres intrinsèques des caméras seront estimés par une procédure d'autocalibrage. Dans la seconde partie, ces paramètres seront estimés par un calibrage par une mire de calibrage planaire.

1.3.2 Capteurs de mouvement

Les capteurs de mouvement mesurent la cinématique d'un système (en translation ou en rotation) sans avoir besoin d'un repère externe (les mesures sont relatives). Ces capteurs proprioceptifs ont l'avantage de délivrer leurs mesures à grande fréquence, typiquement de 30 à 100 Hz, mais informent seulement du mouvement du système et nous n'avons alors aucune information sur la pose initiale du système. Dans cette thèse nous utilisons plusieurs types de capteurs de mouvement : gyroscopes, accéléromètres, magnétomètres (intégrés dans une centrale inertielle) et un odomètre.

1.3.2.1 Gyroscopes

Un gyroscope 3D, constitué de 3 gyroscopes positionnés orthogonalement, mesure la vitesse angulaire ${}^{\mathcal{W}_x}\boldsymbol{\omega}_t$ du système et fournit une observation bruitée de celle-ci, ${}^{\mathcal{W}_x}\hat{\boldsymbol{\omega}}_t$, exprimée dans le repère local du capteur $\mathcal{W}_x$. Pour simplifier les notations, nous omettrons de mentionner le repère dans les formules. La mesure peut être modélisée par l'équation suivante :

$$\hat{\boldsymbol{\omega}}_t = \boldsymbol{\omega}_t + \mathbf{b}_t^{\omega} + \mathbf{n}_t^{\omega}, \tag{1.13}$$

où $\mathbf{b}_t^{\omega}$ est le biais de la mesure (bruit non blanc). Le processus de calibrage du capteur tend à minimiser ce biais, mais il reste toujours un biais à cause des erreurs de calibrage et de la non modélisation de l'intéraction de l'accélération sur la mesure. Le reste des erreurs est modélisé par un bruit blanc Gaussien $\mathbf{n}_t^{\omega}$.

Nous verrons par la suite que cette mesure peut être utilisée pour calculer l'orientation du système par une intégration temporelle t.

1.3.2.2 Accéléromètres

Un accéléromètre 3D est constitué de 3 accéléromètres positionnés orthogonalement. Les technologies employées dans la réalisation des ces capteurs font que celui-ci mesure l'ensemble des accélérations subies par le système, c'est-à-dire l'accélération intrinsèque (ou libre) ${}^{\mathcal{W}_x}\mathbf{a}_t$ et l'accélération due à la gravité terrestre ${}^{\mathcal{W}_x}\mathbf{g}$, exprimées dans le repère du capteur. De la même façon qu'avec les gyroscopes, la mesure d'accélération ${}^{\mathcal{W}_x}\hat{\mathbf{a}}_t$ délivrée par l'accéléromètre 3D est

1. http://phototour.cs.washington.edu/bundler

perturbée par un biais $\mathbf{b}_t^a$ et par un bruit blanc $\mathbf{n}_t^a$ (exprimés aux aussi dans la repère du capteur) :

$$^{\mathcal{W}_x}\hat{\mathbf{a}}_t = {}^{\mathcal{W}_x}\mathbf{a}_t + {}^{\mathcal{W}_x}\mathbf{g} + {}^{\mathcal{W}_x}\mathbf{b}_t^a + {}^{\mathcal{W}_x}\mathbf{n}_t^a. \tag{1.14}$$

Si l'on désire récupérer seulement l'accélération intrinsèque $^{\mathcal{W}_x}\mathbf{a}_t$ du système, nous devons extraire de la mesure $^{\mathcal{W}_x}\hat{\mathbf{a}}_t$ la composante de l'accélération gravitationnelle $^{\mathcal{W}_x}\mathbf{g}$. Or, cette accélération est connue et approximativement constante dans le repère terrestre : $^{\mathcal{W}_g}\mathbf{g}^\top = [0\ 0\ 9,809]$.

$$^{\mathcal{W}_g}\mathbf{a}_t = {}^{\mathcal{W}_g}\mathcal{R}_t^\top {}^{\mathcal{W}_x}\hat{\mathbf{a}}_t - {}^{\mathcal{W}_g}\mathbf{g}. \tag{1.15}$$

Les équations ci-dessus montrent qu'une fois que l'accélération $^{\mathcal{W}_g}\mathbf{a}_t$ est connue, la mesure peut être utilisée pour estimer l'orientation $^{\mathcal{W}_g}\mathcal{R}_t^\top$ ou alternativement lorsque l'orientation $^{\mathcal{W}_g}\mathcal{R}_t^\top$ est connue, l'accélération intrinsèque du système dans le repère global $^{\mathcal{W}_g}\mathbf{a}_t$ peut être estimée (mais bruitée).

Il existe principalement deux technologies pour fabriquer des accéléromètres : les accéléromètres MEMS (microelectricomécanique), et les plus perfectionnés (et les plus chers), les accéléromètres utilisant la technologie des fibres optiques (voir section 1.3.3.2).

Nous verrons par la suite que l'accéléromètre peut être employé pour calculer la vitesse voire la position du système en intégrant une à deux fois la mesure sur le temps t. Les résultats montrent que cette méthode est très affectée par une dérive.

1.3.2.3 Odomètre

L'odomètre est un capteur mesurant la distance parcourue par un système mobile. Il utilise en interne un tachymètre placé sur chaque roue qui mesure leur vitesse de rotation. Il ne peut donc pas être utilisé dans une application de localisation embarquée sur un individu. Grâce à un calibrage précis de la circonférence des roues, l'odomètre peut estimer la distance (en mètres) parcourue. Il est aussi fréquent d'avoir une information de vitesse linéaire (en translation) $^{\mathcal{W}_o}\hat{v}_t^o$, exprimé en kilomètres par heure (km/h) en lieu et place de la distance $^{\mathcal{W}_o}\hat{s}_t^o$.

1.3.2.4 Centrale inertielle (IMU)

La centrale inertielle (*Inertial Measurement Unit*) que nous avons utilisée est la centrale MTi de la société XSENS [Xsens 2008] (voir la figure 1.3). Celle-ci combine différents capteurs inertiels : des gyroscopes, des accéléromètres et des magnétomètres. Chacun de ces capteurs est dupliqué en 3 exemplaires, positionnés orthogonalement aux 3 axes (x,y,z) du repère local de la centrale $\mathcal{W}_x$ et forme ainsi un capteur 3D. La centrale inertielle délivre des données brutes en 3 dimensions : vitesse angulaire (degrés par seconde), accélération tangentielle (mètres par seconde au carré) et champ magnétique (milligauss), ainsi qu'une information d'orientation 3D exprimée dans le repère global $\mathcal{W}_g$.

Les capteurs internes de la centrale sont calibrés (alignement des axes des capteurs, covariances du bruit de mesure) par le constructeur (ou par l'utilisateur) et possède un système de compensation des erreurs dues à la température du capteur. Cependant, le faible coût et la taille réduite de ces capteurs MEMS font que la performance de la méthode est souvent limitée en

FIGURE 1.3 – La centrale inertielle XSENS MTi.

terme de précision et de stabilité du biais sur les mesures, ce qui est le principal facteur de dérive des techniques de navigation à guidage inertiel (INS).

L'orientation 3D est obtenue à partir des données brutes des capteurs internes, par un microprocesseur interne au capteur. Les accéléromètres et les magnétomètres sont utilisés pour compenser la dérive d'orientation lors de l'intégration des données des gyroscopes. De plus les magnétomètres permettent à la centrale inertielle de connaître le Nord magnétique de la Terre et celle-ci peut alors donner l'orientation 3D dans le repère global. Cependant, les magnétomètres sont très sensibles aux matériaux ferromagnétiques et aux champs magnétiques environnants et les mesures peuvent ainsi être perturbées. L'orientation 3D finale estimée par la centrale sera alors elle aussi faussée.

Le capteur MTi de la société XSENS n'estime pas dynamiquement les covariances des observations qu'il délivre. Les écarts types des capteurs internes sont cependant fournit par la société XSENS (voir le tableau 1.1). Les informations envoyées par la centrale inertielle sont les suivantes :

- l'accélération ${}^{\mathcal{W}_x}\hat{\mathbf{a}}_t^x$
- la vitesse angulaire ${}^{\mathcal{W}_x}\hat{\boldsymbol{\omega}}_t^x$
- le champ magnétique ${}^{\mathcal{W}_x}\hat{\mathbf{m}}_t^x$
- l'orientation ${}^{\mathcal{W}_x}\hat{\mathcal{R}}_t^x$.

1.3.3 Capteurs de position

À l'inverse des capteurs proprioceptifs, les capteurs extéroceptifs (de position) utilisent l'environnement extérieur au système mobile, pour délivrer mesurer leurs informations de localisation, souvent à une faible fréquence (quelques Hertz). Dans cette thèse nous avons employé plusieurs capteurs de position : un système de navigation à guidage inertiel (INS), un GPS et un tracker laser.

1.3.3.1 GPS

Le système GPS (*Global Positioning System*) est constitué d'un ensemble de satellites et permet à un récepteur GPS positionné sur Terre de connaître sa position en 3 dimensions. Ce capteur est cependant peu précis (environs 10 mètres) et de faible fréquence ($\sim$ 1Hz). Il n'est donc pas possible d'utiliser ce capteur pour localiser un système en intérieur. Les données provenant des

différents GPS auxquels nous avons eu accès n'ont pas été employées directement, mais intégrées à un système de navigation à guidage inertiel (INS).

Il existe une catégorie de GPS beaucoup plus précis, les GPS-RTK (*Real Time Kinematic*). Une station de référence fixe, localisée près du récepteur, fournit des corrections en temps réel permettant d'atteindre une précision de l'ordre du centimètre. Cette précision peut cependant diminuer fortement suivant plusieurs paramètres comme la qualité des stations de base, l'éloignement du récepteur ou encore les multitrajets dues aux réflexions des ondes sur les bâtiments en milieu urbain.

1.3.3.2 Système de navigation à guidage inertiel (INS)

Principes. La technique d'estimation des position et orientation relatives d'un système mobile en utilisant seulement les gyroscopes et les accéléromètres est appelée navigation à guidage inertiel [Titterton 2004]. Le principe (connu sous le nom anglais de *strapdown*) des systèmes de navigation à guidage inertiel est le suivant : l'orientation est calculée en intégrant selon le temps les mesures des gyroscopes de la centrale inertielle. La position est quant-à-elle estimée en enlevant la gravité des mesures des accéléromètres d'une part, et en intégrant par deux fois les mesures. Ces positions et orientations sont calculées par rapport à une origine. La plupart du temps, les INS intègrent un module GPS ce qui leur permet de localiser un système dans le repère terrestre. Les INS sont principalement utilisés dans les applications nécessitant une localisation précise (de meilleure qualité que le GPS). On les retrouve notamment dans les missiles, les drones, les navires etc.

En pratique, les mesures des centrales inertielles sont affectées par des bruits non blanc. En conséquence, une dérive apparaît sur la position et sur l'orientation calculées par le système. Par exemple, un biais constant sur les mesures des gyroscope entraînera une erreur linéaire sur l'orientation. Un biais constant sur les accélérations introduit une erreur quadratique sur le calcul de position. Avec les accéléromètres de faible coût (MEMS), la dérive de position peut en quelques dizaines de secondes atteindre plusieurs mètres. L'erreur de position est en grande partie due à la mauvaise correction de la gravité sur les mesures d'accélération. En effet, pour corriger la gravité l'INS utilise l'orientation dans le repère monde et si celle-ci est approximative, alors un biais apparaît sur l'accélération, ce qui entraîne une erreur quadratique sur la position finale. Notons que les INS fusionnent différents capteurs (en incluant le GPS) afin de réduire ces dérives [Titterton 2004].

Dans cette thèse nous avons utilisé le système de navigation à guidage inertiel PHINS de la société IXSEA.

Centrale de navigation IXSEA. La centrale de navigation PHINS (figure 1.4) est un système de navigation à guidage inertiel de grande qualité. Elle utilise en interne des accéléromètres et gyroscopes à fibre optique très précis et possède en plus un système de correction des dérives qui fusionne les données de tous les capteurs afin de réduire au maximum les erreurs de localisation. Elle fournit de plus dynamiquement les variances associées aux mesures.

Nous avons eu accès à la centrale IXSEA lors de plusieurs essais, en intérieur et en extérieur. Les informations envoyées par la centrale à une fréquence de 30 Hz sont les suivantes :

– la position ${}^{\mathcal{W}_n}\hat{\mathbf{t}}_t^n$ et les écarts types associés $\sigma_t^{\mathbf{t}}$.
– l'orientation ${}^{\mathcal{W}_n}\hat{\mathcal{R}}_t^n$ (ou ${}^{\mathcal{W}_n}\hat{\mathbf{e}}_t^n$) et les écarts types associés $\sigma_t^{\mathbf{e}}$.

Dans nos expérimentations, la centrale PHINS alliée à un GPS-RTK sera utilisée afin de produire une vérité terrain sur la localisation d'un véhicule en milieu urbain. Elle sera aussi employée seule dans le cadre d'un SLAM multi-capteur (voir chapitre 5).

FIGURE 1.4 – Centrale de navigation à guidage inertiel PHINS de la société IXSEA.

1.3.3.3 Laser de poursuite

Le laser de poursuite (*Laser Tracker* en anglais) est un système de mesure de coordonnées très précis. Le système est composé d'un émetteur laser fixe (interféromètre laser et télémètre absolu) et d'un récepteur (sphère avec des facettes réfléchissantes) fixé sur le système mobile que l'on souhaite localiser. La tête pivotante de l'émetteur poursuit (*tracking*) le récepteur tout au long de la trajectoire et peut ainsi estimer précisément la position 3D du système mobile. Ce type de système a de multiples applications industrielles : mesure de pièces à géométrie complexe, mesure de surface, comparaison au modèle CAO, calibrage de machine-outil et de robots, contrôle d'ensembles mécaniques de grandes dimensions, etc. Nous avons utilisé le tracker Omnitrack de la société Automated Precision Inc. (API), dont la photo est donnée en figure 1.5. La qualité de la localisation par laser de poursuite de ce modèle est donnée à 25 $\mu m \pm 5\mu m$, en conditions optimales. Ce capteur nous fournira certaines mesures très précises (considérées comme la vérité terrain) auxquelles nous comparerons nos algorithmes de localisation. L'inconvénient de ce système est qu'il offre une portée limitée ($40m$), ce qui nous empêche de l'utiliser sur des expérimentations de longues distances.

Le tracker laser délivre les positions à une fréquence de 50 Hz, exprimées dans le repère local du tracker laser $\mathcal{W}_{laser}$. Les informations de sortie auxquelles nous avons eu accès étaient directement exprimées dans le repère terrestre $\mathcal{W}_g$, grâce à un recalage *a posteriori*. Les positions de sortie sont notées ${}^{\mathcal{W}_g}\hat{\mathbf{t}}_t^{laser}$.

FIGURE 1.5 – Laser de poursuite Omnitrack de la société API.

1.3.4 Récapitulatif

Nous résumons l'ensemble des capteurs utilisés dans cette thèse dans le tableau 1.1 suivant :

Capteur	Type (section)	Freq.	Données	Unité	Design.	Précision (σ)
AVT Guppy	Caméra (1.3.1)	30 Hz	image NdG 640×480	$pixels$	I^c	-
Odomètre	Odomètre (1.3.2.3)	25 Hz	distance ou vitesse	$m/m.s^{-1}$	$\hat{s}^o/\hat{v}^o$	-
XSENS MTi	IMU (1.3.2.4)	100 Hz	accélération 3D	$m.s^{-2}$	$\hat{\mathbf{a}}^x$	0.02
			vitesse angulaire 3D	$deg.s^{-1}$	$\hat{\omega}^x$	0.1
			champ magnétique 3D	$mGauss$	$\hat{\mathbf{m}}^x$	0.5
			température	°C	$\hat{T}^x$	0.5
			orientation 3D	-	$\hat{\mathcal{R}}^x$	$0.5°$[1], $1°$[2]
IXSEA PHINS	INS (1.3.3.2)	100 Hz	position 3D	m	$\hat{\mathbf{t}}^n$	$3.10^{-4}m.s^{-2}$[3]
			orientation 3D	-	$\hat{\mathcal{R}}^n$	$0.01°$
API Omnitrack	*Laser tracker* (1.3.3.3)	50 Hz	position 3D	m	$\hat{\mathbf{t}}^{laser}$	$25\mu m \pm 5\mu m$

TABLE 1.1 – Nomenclature des capteurs utilisés.

Freq : fréquence d'acquisition des mesures. NdG : Niveaux de gris. Design. : désignation.

1.4 Mesures de distance

Dans nos travaux nous avons besoin de mesurer la qualité de la localisation et de la reconstruction de la scène obtenue. Pour cela, il nous est nécessaire d'introduire un critère de qualité que l'on associera à la distance entre deux points : le point observé $\hat{\mathbf{q}}$ et le point prédit par le

1. Précision statique en roulis et tangage (2° en dynamique)
2. Précision statique en cap (2° en dynamique)
3. ou $\frac{\sigma_{GPS}}{3}$ lorsque le GPS est accessible

modèle $\mathbf{q}$. Le terme "point" pourra représenter la position 3D du centre optique d'une caméra ou d'un point 3D de la scène, mais aussi une position 2D d'un point d'intérêt dans l'image.

Nous introduisons plusieurs distances entre deux points : la distance géométrique Euclidienne, la distance statistique de Mahalanobis et une distance algébrique.

1.4.1 Distance Euclidienne

La distance Euclidienne mesure la distance géométrique Euclidienne entre deux points exprimés en coordonnées homogènes $\hat{\mathbf{q}} = \begin{bmatrix} \tilde{\hat{\mathbf{q}}} \\ \hat{w} \end{bmatrix}$ et $\mathbf{q} = \begin{bmatrix} \tilde{\mathbf{q}} \\ w \end{bmatrix}$ est notée $d(\hat{\mathbf{q}}, \mathbf{q})$ et est définie par :

$$d(\hat{\mathbf{q}}, \mathbf{q}) = \left\| \Psi(\hat{\mathbf{q}}) - \Psi(\mathbf{q}) \right\|_2 = \left\| \frac{\tilde{\hat{\mathbf{q}}}}{\hat{w}} - \frac{\tilde{\mathbf{q}}}{w} \right\|_2 . \tag{1.16}$$

Notons que la distance Euclidienne est non linéaire en ses paramètres. Elle est aussi non quadratique en raison de la fonction de passage des coordonnées homogènes en coordonnées non-homogènes Ψ (et de la racine).

1.4.2 Distance de Mahalanobis

La distance de Mahalanobis est une mesure de distance statistique qui informe de la similarité entre deux vecteurs. À la différence de la distance Euclidienne où toutes les composantes des vecteurs sont traitées de la même façon, la distance de Mahalanobis accorde un poids moins important aux composantes de peu de confiance. Cela implique l'existence d'un critère de confiance associé soit à la mesure entière, soit à une des composantes de la mesure. Dans le cas général, la confiance d'une observation est modélisée par une matrice de variance-covariance, notée Σ, qui informe de la confiance et de la corrélation entre les différentes composantes de l'observation. La distance de Mahalanobis sous sa forme générale s'écrit :

$$d_{\Sigma}(\hat{\mathbf{q}}, \mathbf{q}) = \left\| \Psi(\hat{\mathbf{q}}) - \Psi(\mathbf{q}) \right\|_{\Sigma} , \tag{1.17}$$

où la norme de Mahalanobis est $\|\mathbf{x} - \mathbf{y}\|_{\Sigma} = \sqrt{(\mathbf{x} - \mathbf{y})^{\top} \Sigma^{-1} (\mathbf{x} - \mathbf{y})}$.

En pratique, il est courant de n'associer qu'une seule valeur de confiance σ_i pour chaque composante i, la matrice Σ est alors diagonale. La distance de Mahalanobis ainsi configurée est aussi appelée distance Euclidienne normalisée et s'écrit :

$$d_{\Sigma}(\hat{\mathbf{q}}, \mathbf{q}) = \sum_{i=1}^{p} \frac{1}{\sigma_i^2} \left(\frac{\tilde{\hat{q}}_i}{\hat{w}} - \frac{\tilde{q}_i}{w} \right). \tag{1.18}$$

Notons que la distance Euclidienne correspond à la distance de Mahalanobis avec une matrice de variance-covariance identité : $d(\hat{\mathbf{q}}, \mathbf{q}) = d_{\mathcal{I}}(\hat{\mathbf{q}}, \mathbf{q})$.

1.4.3 Distance algébrique

La distance algébrique [Hartley 2003b, p. 89] est une pseudo-distance généralement employée afin d'obtenir une solution simple du problème. En effet, sous certains conditions, cette distance se veut être une approximation quadratique de la distance géométrique. Une distance algébrique entre deux points $\hat{\mathbf{q}}$ et $\mathbf{q}$ peut être définie à partir du produit vectoriel des deux vecteurs de coordonnées. En effet, si les points sont identiques, alors le produit vectoriel ainsi formé sera nul. Dans le cas échéant, ce produit sera non nul et nous pouvons dès lors construire une pseudo-distance (qui ne respecte pas l'inégalité triangulaire) à partir de celui-ci. Nous définissons une distance algébrique $d_{\mathcal{A}}(\hat{\mathbf{q}}, \mathbf{q})$ de la manière suivante :

$$d_{\mathcal{A}}(\hat{\mathbf{q}}, \mathbf{q}) = \|\mathcal{S}[\hat{\mathbf{q}}]_{\times}\mathbf{q}\|\,, \tag{1.19}$$

où $[\mathbf{q}]_{\times}$ est la matrice associée au produit vectoriel et $\mathcal{S} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ est une matrice permettant de ne sélectionner que les deux premières lignes du produit vectoriel (puisque seulement deux lignes sont linéairement indépendantes).

Dans la suite de ce mémoire, nous verrons que cette distance possède l'avantage de simplifier le problème de l'ajustement de faisceaux, un problème non linéaire non quadratique à cause de la distance géométrique employée.

CHAPITRE 2

Estimation de paramètres et sélection de modèle

Dans ce chapitre nous introduisons les outils mathématiques qui seront utilisés dans la suite de la thèse. Dans un premier temps, nous nous intéressons au problème inverse pour retrouver le modèle et ses paramètres à partir d'observations. Puis, nous présentons les techniques d'optimisation non-linéaire non contrainte adaptées aux fonctions de coût de moindres carrés, ainsi que les méthodes d'optimisation robuste. Enfin, nous abordons le problème de la sélection du modèle et des techniques associées.

In this chapter we introduce the mathematical tools that will be used later in the thesis. At first, we investigate the inverse problem to find the parameters of a model from its observations. Then, we present unconstrained optimization techniques suitable for linear or nonlinear least squares problems and robust optimization and multi-objective methods. Finally, we address the problem of model selection and related techniques.

2.1 Formalisation du problème

Dans cette thèse nous explorons le sujet de la localisation par la vision dont l'objectif est de retrouver la trajectoire du système $\mathbf{x}^{\mathcal{P}}$ et la structure de l'environnement $\mathbf{x}^{Q}$. Il s'agit d'un problème inverse dans lequel nous cherchons à estimer les paramètres $\mathbf{x}$ du modèle sans *a priori* sur l'environnement, seulement à partir des observations $\hat{\mathbf{z}}$ fournies par les différents capteurs.

Si les observations étaient parfaites, les paramètres pourraient être estimés avec une qualité infinie à partir d'un minimum d'observations : n observations suffiraient à estimer n paramètres du modèle [1]. Mais les observations sont généralement bruitées et l'on doit alors combiner un maximum de mesures pour estimer avec précision le modèle.

Pour cela, nous modélisons le processus d'observation de chaque capteur k par la fonction $h_t^k(\mathbf{x}_t)$, comme spécifié dans l'équation (1.4). Ces fonctions d'observation, par exemple la projection de la scène dans une image, seront considérées connues et nous nous intéressons alors au problème de l'estimation des paramètres du modèle. Nous considérons que la fonction de vraisemblance de chaque observation $\hat{z}$ et que le processus d'observation des capteurs sont modélisés par une variable aléatoire suivant une loi normale gaussienne, telle que :

$$p(\hat{z}|\mathbf{x}) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}\left(\frac{\hat{z}-h(\mathbf{x})}{\sigma}\right)^2}. \tag{2.1}$$

La théorie de l'estimation nous informe que les paramètres $\mathbf{x}$ optimaux que l'on recherche sont ceux maximisant l'ensemble des fonctions de vraisemblance de toutes les observations de l'ensemble des capteurs, regroupées dans le vecteur d'observation $\hat{\mathbf{z}} = [\ldots \hat{z}_i \ldots]$. L'objectif est alors de trouver les paramètres $\mathbf{x}$ qui maximisent la probabilité d'avoir mesuré l'ensemble des observations :

$$\max_{\mathbf{x}\in\mathfrak{M}} p(\mathbf{x}|\hat{\mathbf{z}}). \tag{2.2}$$

Ce problème est généralement réécrit à l'aide du théorème de Bayes ($p(\mathbf{x}|\hat{\mathbf{z}}) = \frac{p(\hat{\mathbf{z}}|\mathbf{x})p(\mathbf{x})}{p(\hat{\mathbf{z}})}$) et résolu (en éliminant le terme $p(\hat{\mathbf{z}})$, constant par rapport à $\mathbf{x}$), avec un estimateur au maximum de vraisemblance ou au maximum *a posteriori*. Les estimateurs au maximum de vraisemblance (ML : *Maximum Likelihood*) s'intéressent à maximiser la probabilité $p(\hat{\mathbf{z}}|\mathbf{x})$:

$$\max_{\mathbf{x}\in\mathfrak{M}} p(\hat{\mathbf{z}}|\mathbf{x}), \tag{2.3}$$

où $p(\hat{\mathbf{z}}|\mathbf{x})$ est la fonction de densité de probabilité des observations $\hat{\mathbf{z}}$ avec les paramètres du modèle $\mathbf{x}$. Lorsque la densité de probabilité $p(\mathbf{x})$ des paramètres du modèle est connue *a priori*, nous pouvons intégrer celle-ci et nous obtenons alors un estimateur maximum *a posteriori* (MAP) :

$$\max_{\mathbf{x}\in\mathfrak{M}} p(\hat{\mathbf{z}}|\mathbf{x})p(\mathbf{x}). \tag{2.4}$$

Le problème (2.3) peut être simplifié lorsque les densités de probabilité sont gaussiennes et peut se réécrire sous la forme d'une minimisation en prenant le logarithme de $p(\hat{\mathbf{z}}|\mathbf{x})$, noté $F(\mathbf{x}) \sim -\log p(\hat{\mathbf{z}}|\mathbf{x})$:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathfrak{M}} p(\hat{\mathbf{z}}|\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathfrak{M}} F(\mathbf{x}). \tag{2.5}$$

1. Excepté la précision numérique (arrondis) et les singularités du problème.

Sous l'hypothèse d'indépendance des erreurs de mesure, la fonction de probabilité des paramètres du modèle avec l'ensemble des observations peut se décomposer en un produit des probabilités de chaque observation :

$$p(\hat{\mathbf{z}}|\mathbf{x}) = \prod_{i=1}^{N_z} p_i(\hat{z}_i|\mathbf{x}). \tag{2.6}$$

Sous l'hypothèse que ces erreurs suivent un modèle de bruit gaussien centré ($\mathcal{N}(0, \sigma^2)$), pour toute valeur de variance $\sigma^2 \in \mathbb{R}$, nous pouvons montrer que la fonction de densité $p(\hat{\mathbf{z}}|\mathbf{x})$ est maximisée quand la somme des carrés des résidus est minimisée. La fonction de coût $F(\mathbf{x})$: $\mathfrak{M} \to \mathbb{R}^{N_z}$, parfois notée $\chi^2_\Sigma(\mathbf{x})$ en référence à la loi statistique qu'elle décrit, s'écrit dans ce cas :

$$F(\mathbf{x}) = \chi^2_\Sigma(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{N_z}\left(\frac{\hat{z}_i - h_i(\mathbf{x})}{\sigma_i}\right)^2 \sim -\log p(\hat{\mathbf{z}}|\mathbf{x}). \tag{2.7}$$

Les erreurs quadratiques peuvent être mises sous la forme d'un produit de vecteur, tel que :

$$\chi^2_\Sigma(\mathbf{x}) = \frac{1}{2}\boldsymbol{\Delta}(\mathbf{x})^\top \Sigma^{-1}\boldsymbol{\Delta}(\mathbf{x}), \tag{2.8}$$

où $\Sigma = \mathrm{diag}(\ldots, \sigma_i^2, \ldots)$ est la matrice diagonale formée des variances des mesures (où plus généralement une matrice symétrique) et $\boldsymbol{\Delta}(\mathbf{x})$ est le vecteur contenant tous les résidus :

$$\boldsymbol{\Delta}(\mathbf{x}) = \hat{\mathbf{z}} - h(\mathbf{x}) = \begin{bmatrix} \cdots \\ \Delta_i(\mathbf{x}) \\ \cdots \end{bmatrix} \quad \text{avec} \quad \Delta_i(\mathbf{x}) = \hat{z}_i - h_i(\mathbf{x}). \tag{2.9}$$

La fonction de coût χ^2_Σ est par conséquent constituée des résidus pondérés par les variances associées, aussi connu sous le nom de distance de Mahalanobis entre les observations et les valeurs prédites par le modèle :

$$\chi^2_\Sigma(\mathbf{x}) = \frac{1}{2}\|\boldsymbol{\Delta}(\mathbf{x})\|^2_\Sigma. \tag{2.10}$$

Plus simplement, dans le cas où toutes les observations ont la même incertitude d'écart-type σ nous avons $\Sigma = \sigma^2\mathcal{I}$ et

$$\chi^2_\Sigma(\mathbf{x}) = \frac{1}{2\sigma^2}\|\boldsymbol{\Delta}(\mathbf{x})\|^2_2 = \frac{1}{2\sigma^2}\sum_{i=1}^{N_z}\|\hat{\mathbf{z}}_i - h_i(\mathbf{x})\|^2 \tag{2.11}$$

Le problème de l'estimation des paramètres du modèle revient donc dans notre situation à minimiser la fonction de coût $F(\mathbf{x})$ (équation (2.5)), constituée de la somme des carrés des résidus entre le modèle et les observations. Ce problème peut être traité à l'aide des techniques de moindres carrés.

2.2 Méthodes de moindres carrés

Les méthodes de moindres carrés s'intéressent à résoudre les problèmes dont la fonction de coût $F(\mathbf{x}) : \mathfrak{M} \to \mathbb{R}^{N_z}$ prend la forme de l'équation (2.7). L'objectif des méthodes de moindres carrés est de rechercher les paramètres $\mathbf{x}$ optimaux dans l'espace $\mathfrak{M}$, qui minimisent la fonction objectif $F(\mathbf{x})$:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathfrak{M}} F(\mathbf{x}) = \frac{1}{2}\|\hat{\mathbf{z}} - h(\mathbf{x})\|_2^2. \tag{2.12}$$

La norme des résidus pourra être une norme pondérée (en utilisant une distance de Mahalanobis : $\|\hat{\mathbf{z}} - h(\mathbf{x})\|_\Sigma$) et nous parlerons alors de moindres carrés pondérés.

Nous appelons le point $\mathbf{x}^*$ minimum global de la fonction F si $F(\mathbf{x}^*) \leq F(\mathbf{x})$ pour tout $\mathbf{x} \in \mathfrak{M}$. Pour les fonctions de coût non-linéaires (donc non convexes), le minimum global de la fonction de coût peut être difficile, voire impossible pratiquement à trouver car notre connaissance de la fonction est souvent locale. Il est en effet généralement impossible de visiter l'ensemble de la variété pour trouver ce point particulier. Aussi, un point $\mathbf{x}^*$ sera dit minimum local s'il existe un voisinage $M \in \mathfrak{M}$ de $\mathbf{x}^*$ tel que $F(\mathbf{x}^*) \leq F(\mathbf{x})$ pour $\mathbf{x} \in M$.

Avant de présenter les techniques itératives résolvant les problèmes de moindres carrés non-linéaires, nous considérons tout d'abord le cas où la fonction d'observation $h(\mathbf{x})$ est linéaire.

2.2.1 Moindres carrés linéaires

2.2.1.1 Résolution

Lorsque les observations $\hat{\mathbf{z}}$ constituent des combinaisons linéaires des paramètres $\mathbf{x}$ du modèle, la fonction d'observation linéaire $h(\mathbf{x})$ peut être modélisée par une matrice que l'on notera $\mathcal{L}$:

$$h(\mathbf{x}) = \mathcal{L}\mathbf{x}, \tag{2.13}$$

et la fonction de coût F devient dès lors quadratique (et la fonction $\boldsymbol{\Delta}(\mathbf{x})$ linéaire) et s'écrit :

$$F(\mathbf{x}) = \frac{1}{2}\|\hat{\mathbf{z}} - \mathcal{L}\mathbf{x}\|_2^2. \tag{2.14}$$

La solution de ce problème quadratique (donc convexe) peut être obtenue directement en résolvant les équations normales :

$$\mathbf{x}^* = (\mathcal{L}^\top\mathcal{L})^{-1}\mathcal{L}\hat{\mathbf{z}} = \mathcal{L}^\dagger\hat{\mathbf{z}}, \tag{2.15}$$

où $\mathcal{L}^\dagger$ est la pseudo-inverse de MOORE–PENROSE de $\mathcal{L}$.

Il est à noter l'existence d'autres méthodes plus stables pour résoudre ce type de problème, utilisant par exemple une décomposition SVD (*Singular Value Decomposition*), de CHOLESKY ou QR [Björk 1996].

2.2.1.2 Régularisation

La résolution du problème précédent peut ne pas aboutir à une solution unique du système, par exemple lorsque la matrice $\mathcal{L}$ est mal conditionnée ou singulière. Le problème est alors mal posé. Il faut utiliser dans ce cas une régularisation, par exemple la régularisation de TIKHONOV ou régression d'arête (*ridge regression*). Ce principe de régularisation consiste à ajouter un terme de régularisation modélisé par une matrice $\mathcal{D}$, directement dans la fonction de coût, afin de privilégier une parmi les multiples solutions. La nouvelle fonction objectif s'écrit :

$$F(\mathbf{x}) = \frac{1}{2}\|\hat{\mathbf{z}} - \mathcal{L}\mathbf{x}\|_2^2 + \frac{1}{2}\|\mathcal{D}\mathbf{x}\|^2 \tag{2.16}$$

La solution stable de ce nouveau système est alors :

$$\mathbf{x}^* = (\mathcal{L}^\top\mathcal{L} + \mathcal{D}^\top\mathcal{D})^{-1}\mathcal{L}\hat{\mathbf{z}}. \tag{2.17}$$

Le résultat de cette régularisation va bien entendu dépendre du choix de la matrice $\mathcal{D}$. Une solution possible est de prendre la forme $\mathcal{D} = \lambda\mathcal{I}$, où $\lambda \in \mathbb{R}^+$ représentera l'importance de la régularisation de la solution souhaitée (lorsque $\lambda = 0$, on retrouve la solution (2.15)). Par la suite nous verrons qu'il existe différentes manières de sélectionner automatiquement la valeur du paramètre de régularisation λ à partir des données du problème (voir section la 2.4).

2.2.2 Moindres carrés non-linéaires

La fonction de coût du problème est souvent non-linéaire en ses paramètres et il est alors nécessaire d'employer d'autres techniques pour trouver le minimum. La résolution des problèmes de moindres carrés non-linéaires est un problème difficile dans le sens où il est généralement impossible de parcourir l'ensemble de l'espace des paramètres, et que la fonction objectif peut présenter de nombreux minima locaux. Il est nécessaire d'adopter un comportement incrémental dans lequel plusieurs minimisations locales sont effectuées. Ce processus génère des déplacements à chaque itération k dans l'espace local des paramètres.

Paramétrisation locale des paramètres. Comme nous l'avons précisé au début du chapitre, l'espace formé par les N_p paramètres du modèle est l'espace $\mathfrak{M}$ (de $\mathbb{R}^{N_p}$). Le vecteur des paramètres $\mathbf{x}$ contient en partie les orientations du système. Celles-ci sont paramétrées par des matrices orthogonales de taille 3×3 et forment le groupe orthogonal SO(3). Or, les méthodes de moindres carrés de l'état de l'art reposent sur des algorithmes d'optimisation souvent non contrainte, c'est-à-dire dans l'espace Euclidien $\mathbb{R}$. Par conséquent, nous devons étendre l'espace local au point $\mathbf{x}^{(k)}$ de la variété $\mathfrak{M}$ par son espace affine tangent, à l'aide d'une projection dans $\mathbb{R}^{N_d}$ où $N_d \leq N_p$. Nous introduisons pour cela la paramétrisation locale des paramètres du modèle $\mu_k(\boldsymbol{\delta}^{(k)}) \in \mathfrak{M}$ autour de $\mathbf{x}^{(k)}$ suivante :

$$(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)}) \in \mathfrak{M} \times \mathbb{R}^{N_d} \rightarrow \mu_k(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)}) \in \mathfrak{M}, \tag{2.18}$$

où $\boldsymbol{\delta}$ est un petit vecteur de déplacement constitué de N_d coefficients. La fonction de mise à jour d'une itération est alors notée :

$$F(\mathbf{x}^{(k+1)}) = F(\mu_k(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)})) = F_{\mu_k}(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)}). \tag{2.19}$$

Les matrices de rotation du groupe orthogonal SO(3) sont ainsi paramétrées de façon locale par exemple par 3 angles d'Euler $\boldsymbol{\delta}_e^{(k)}$:

$$\mathbf{x}^{(k+1)} = \mu_k(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)}) : \mathcal{R}_i^{(k+1)} = \mathcal{R}_i^{(k)} \, \mathrm{rot_e}\left(\boldsymbol{\delta}_e^{(k)}\right) \tag{2.20}$$

Les autres paramètres du modèle, comme les positions, sont paramétrés par leurs coordonnées homogènes ou non homogènes. La paramétrisation locale est dans notre application nécessaire puisque le vecteur des paramètres $\mathbf{x}$ contient des orientations. Nous verrons par la suite que cette paramétrisation locale n'est pas nécessaire dans les ajustements de faisceaux non calibrés. Dans un tel cas, le déplacement des paramètres du modèles se réalise par une simple addition vectorielle :

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \boldsymbol{\delta}^{(k)} \tag{2.21}$$

L'idée ici est donc de simplifier l'espace des paramètres du modèle pour être dans $\mathbb{R}^{N_d}$, d'appliquer ensuite les méthodes de moindres carrés et optimiser sur les paramètres locaux de déplacement, puis de revenir dans l'espace $\mathfrak{M}$ par la projection inverse.

Approximation locale de la fonction de coût, matrices jacobienne et hessienne. L'idée générale des techniques de minimisation non-linéaire au sens des moindres carrés est de simplifier le problème, en remplaçant la fonction objectif $F(\mathbf{x})$, non-linéaire par définition, par une approximation locale e $\mathbf{x}$ notée m, paramétrée par un petit déplacement $\boldsymbol{\delta} \in \mathbb{R}^{N_p}$ dans l'espace tangent de $\mathfrak{M}$. Nous définissons $m : \mathbb{R}^{N_p} \to \mathbb{R}^{N_z}$ et

$$F_{\mu_k}(\mathbf{x}^{(k)}, \boldsymbol{\delta}) \approx m_k(\boldsymbol{\delta}), \tag{2.22}$$

où $\mathbf{x}^{(k)}$ est un point quelconque de l'espace $\mathfrak{M}$. Pour un petit déplacement $\boldsymbol{\delta} \in \mathbb{R}^{N_p}$ de l'espace tangent au point $\mathbf{x}^{(k)} \in \mathfrak{M}$, la fonction $m_k(\boldsymbol{\delta})$ approxime la fonction de coût du problème initial. Nous remplaçons ainsi le problème d'optimisation non-linéaire du modèle $\mathbf{x}$ sur la variété $\mathfrak{M}$, par plusieurs optimisations non contraintes locales (linéaire ou quadratique suivant l'approximation m) de $\boldsymbol{\delta}$ dans $\mathbb{R}^{N_d}$. Nous verrons que cette approximation locale peut prendre différentes formes suivant les méthodes employées.

Les techniques de minimisation au sens des moindres carrés partent d'une première estimation des paramètres du modèle, que l'on notera $\mathbf{x}^{(0)}$, et se déplacent itérativement dans l'espace des paramètres en résolvant pour chaque déplacement $\boldsymbol{\delta}^{(k)}$ le sous-problème local (2.23) :

$$\boldsymbol{\delta}^\star = \arg \min_{\boldsymbol{\delta} \in \mathbb{R}^{N_p}} m_k(\boldsymbol{\delta}). \tag{2.23}$$

La série de déplacement $\{\ldots, \boldsymbol{\delta}^{(k)}, \ldots\}$ est alors construite de telle sorte qu'elle converge vers un point stationnaire ou minimum $\mathbf{x}^*$ de la fonction F.

La fonction objectif F peut être approximée en une série de TAYLOR. Par exemple, l'approximation $m(\boldsymbol{\delta})$ du premier ordre de $F(\mathbf{x})$ sur le voisinage de $\mathbf{x}$ est :

$$F_{\mu_k}(\mathbf{x}^{(k)}, \boldsymbol{\delta}) \approx m_k(\boldsymbol{\delta}) = F(\mathbf{x}) + \mathcal{J}_F(\mathbf{x}, \boldsymbol{\delta})\boldsymbol{\delta}, \tag{2.24}$$

où $\mathcal{J}_F(\mathbf{x}, \boldsymbol{\delta})$ est la matrice jacobienne de la fonction de coût qui contient les dérivées partielles de F par rapport au vecteur de déplacement $\boldsymbol{\delta}$, dans l'espace tangent au point $\mathbf{x}$. Dans notre problème de moindres carrés, les résidus sont approximés à l'aide de notre connaissance de la fonction d'observation $h(\mathbf{x})$. On définit la matrice jacobienne des résidus $\mathcal{J}_\Delta$ de taille $N_d \times N_z$ suivante :

$$\mathcal{J}_\Delta(\mathbf{x}, \boldsymbol{\delta}) = \begin{bmatrix} \frac{\partial \Delta_1}{\partial \delta_1}(\mathbf{x}) & \dots & \frac{\partial \Delta_{N_z}}{\partial \delta_1}(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial \Delta_1}{\partial \delta_{N_d}}(\mathbf{x}) & \dots & \frac{\partial \Delta_{N_z}}{\partial \delta_{N_d}}(\mathbf{x}) \end{bmatrix}. \tag{2.25}$$

Pour simplifier les notations on écrira $\mathcal{J}^{(k)} = \mathcal{J}_\Delta(\mathbf{x}^{(k)}, \boldsymbol{\delta})$ où $\mathbf{x}^{(k)}$ est l'estimation des paramètres à l'itération k. Remarquons que la matrice Jacobienne des résidus correspond finalement à la matrice jacobienne de la fonction d'observation : $\mathcal{J}_\Delta = \mathcal{J}_h$ car les mesures $\hat{\mathbf{z}}$ sont constantes par rapport aux paramètres. Cette matrice Jacobienne nous permet aussi de calculer le gradient des résidus : $\boldsymbol{\nabla}_\Delta^{(k)} = \mathcal{J}_\Delta^{(k)} \boldsymbol{\Delta}(\mathbf{x}^{(k)})$.

Une autre matrice qui peut être employée dans l'approximation (du second ordre) des résidus est la matrice Hessienne :

$$\mathcal{H}_\Delta(\mathbf{x}, \boldsymbol{\delta}) = \begin{bmatrix} \frac{\partial^2 \Delta_1}{\partial^2 \delta_1 \delta_1}(\mathbf{x}) & \dots & \frac{\partial^2 \Delta_{N_z}}{\partial^2 \delta_1 \delta_{N_d}}(\mathbf{x}) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \Delta_1}{\partial^2 \delta_{N_d} \delta_1}(\mathbf{x}) & \dots & \frac{\partial^2 \Delta_{N_z}}{\partial^2 \delta_{N_d} \delta_{N_d}}(\mathbf{x}) \end{bmatrix}. \tag{2.26}$$

Cette matrice comprend les dérivées secondes du vecteur de résidus par rapport au déplacement $\boldsymbol{\delta}$ au point $\mathbf{x}$. La matrice Hessienne est assez complexe à calculer puisqu'il faut estimer toutes les dérivées secondes du vecteur de résidus. Par la suite nous verrons que la majorité des techniques de moindres carrés non-linéaires sont basées sur une approximation de celle-ci : $\mathcal{H}_\Delta^{(k)} \approx \mathcal{B}_\Delta^{(k)} = {\mathcal{J}_\Delta^{(k)}}^\top \mathcal{J}_\Delta^{(k)}$.

Précisons enfin que dans la variante du problème dite de moindres carrés pondérés, la pondération des résidus peut se faire simplement via une matrice de pondération Σ :

$$\begin{cases} \mathcal{J}_\Delta \\ \boldsymbol{\nabla}_\Delta \\ \mathcal{B}_\Delta \end{cases} \quad \text{devient} \quad \begin{cases} \mathcal{J}_\Delta \Sigma^{-1} \\ \mathcal{J}_\Delta \Sigma^{-1} \boldsymbol{\Delta} \\ \mathcal{J}_\Delta^\top \Sigma^{-1} \mathcal{J}_\Delta. \end{cases} \tag{2.27}$$

Recherches linéaires et régions de confiances. Comme nous venons de le voir, ces techniques d'optimisation non-linéaire estiment donc une série de déplacements locaux à entreprendre dans le but de converger vers la solution, le minimum global $\mathbf{x}^*$ dans le meilleur des cas. Nous n'avons pas encore fait mention de la longueur (ou amplitude) des déplacements, notées $\alpha^{(k)}$. Il existe

deux familles de techniques d'optimisation non-linéaire intégrant la longueur du déplacement : les techniques de recherche linéaire (*Line Search* en anglais) ou de région de confiance (*Trust Region*). La principale différence entre ces deux familles réside dans l'ordre par lequel elles estiment la direction et l'amplitude du déplacement. Les techniques de recherche linéaire estiment en premier lieu une direction dans l'espace des paramètres, puis proposent une amplitude pour le déplacement. À l'inverse, les méthodes dites des régions de confiances choisissent l'amplitude maximale autorisée, notée $\Gamma^{(k)}$ et connue sous le nom de rayon de confiance, et calculent ensuite la direction et le meilleur déplacement (amplitude incluse) sous la contrainte de l'amplitude maximale. L'algorithme 1 présente l'algorithme général des techniques de résolution de moindres carrés non-linéaires par une recherche linéaire.

Algorithme 1 : `NonLinearLeastSquaresLineSearch` : Résolution de moindres carrés non-linéaires par recherche linéaire

- Données : Les observations $\hat{\mathbf{z}}$
- Entrées : Les paramètres initiaux du modèle $\mathbf{x}^{(0)}$
- Sorties : Les paramètres du modèle optimaux $\mathbf{x}^*$, point stationnaire de la fonction de coût F

Initialiser l'itérateur $k \leftarrow 0$;
tant que *les critères d'arrêt ne sont pas atteint* **faire**
  Calculer l'approximation locale de la fonction objectif $m_k(\boldsymbol{\delta})$;
  Estimer un déplacement $\boldsymbol{\delta}^{(k)} \in \mathbb{R}^{N_d}$, tel que $\boldsymbol{\delta}^{(k)} = \arg\min_{\boldsymbol{\delta} \in \mathbb{R}^{N_d}} m_k(\boldsymbol{\delta})$;
  Optionnel : Proposer une amplitude du déplacement $\alpha^{(k)} \in \mathbb{R}$ telle que $\alpha^{(k)} = \arg\min_{\alpha \in \mathbb{R}} m_k(\alpha \boldsymbol{\delta}^{(k)})$, et modifier le déplacement $\boldsymbol{\delta}^{(k)} \leftarrow \alpha^{(k)} \boldsymbol{\delta}^{(k)}$;
  Estimer la nouvelle solution $\mathbf{x}' \leftarrow \mu(\mathbf{x}^{(k)}, \boldsymbol{\delta}^{(k)})$;
  si *les critères d'acceptation du pas sont atteint* **alors**
    Accepter le déplacement $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}'$;
  sinon
    Rejeter le déplacement $\mathbf{x}^{(k+1)} \leftarrow \mathbf{x}^{(k)}$;
  Incrémenter l'itérateur $k \leftarrow k + 1$;

Critères d'arrêt de l'algorithme. Comme nous venons de le voir, le principe de ces méthodes est de générer une série de déplacements dans l'espace tangent des paramètres du modèle. Le nombre de déplacements peut ne pas être fixé *a priori* et nous devons alors définir quand l'algorithme doit s'arrêter. Le principal critère d'arrêt utilisé est basé sur l'amélioration apportée par l'itération en cours : $\varepsilon_k = F(\mathbf{x}^{(k)}) - F(\mathbf{x}^{(k-1)})$. Nous pouvons définir une amélioration minimale ε_{min} en dessous de laquelle nous considérons que l'algorithme a atteint un point stationnaire (par exemple 10^{-10}). Cela permet ainsi d'éviter à l'algorithme non seulement d'itérer indéfiniment alors que le gain apporté est très réduit, mais aussi de préciser l'*epsilon machine*, c'est-à-dire la plus petite variation que l'ordinateur peut manipuler. Il existe dans la littérature d'autres critères

d'arrêts notamment sur la valeur du gradient [Nocedal 1999].

Critères d'acceptation du pas. Les algorithmes de minimisation non-linéaire proposent des déplacements dans l'espace tangent des paramètres du modèle. Cependant ces déplacements, calculés à partir de l'approximation locale de la fonction de coût, peuvent ne pas améliorer la solution courante et ainsi empêcher la convergence de la méthode. Pour éviter cela, il est courant de filtrer les déplacements incorrects par des critères d'acceptation du pas. Un critère simple est basé sur l'amélioration courante. Si l'erreur du point proposé est supérieure à l'erreur précédente ($F(\mathbf{x}') > F(\mathbf{x}^{(k)})$) alors le déplacement n'est pas effectué et l'algorithme change alors ses paramètres internes ou s'arrête simplement.

Comme nous l'avons précisé, la résolution des problèmes de moindres carrés non-linéaires passe par une approximation locale de la fonction objectif non-linéaire. Cette approximation diffère selon les méthodes employées mais il est courant d'employer l'approximation de TAYLOR, soit du premier ordre, comme par exemple dans la technique de descente de gradient, soit du second ordre ou quadratique dans les méthodes de GAUSS-NEWTON et dérivées.

2.2.2.1 Descente de gradient et gradient conjugué

Descente de gradient. Le principe général de la méthode de descente de gradient (*steepest descent* en anglais) est de choisir à chaque itération un déplacement $\boldsymbol{\delta}_{DG}^{(k)}$ dans la direction opposée au gradient $\boldsymbol{\nabla}^{(k)}$ de la fonction de coût.

$$\boldsymbol{\delta}_{DG}^{(k)} = -\boldsymbol{\nabla}^{(k)}. \tag{2.28}$$

Cette technique de minimisation est l'une des plus simples et souffre de quelques problèmes de rapidité de convergence notamment près du minimum de la fonction objectif.

D'autres techniques utilisent néanmoins l'idée de descente de gradient et améliorent considérablement la rapidité de convergence de la minimisation. C'est le cas de la technique du gradient conjugué.

Gradient conjugué. La méthode du gradient conjugué non-linéaire [Hestenes 1952] résout le problème de minimisation non-linéaire en proposant itérativement des déplacements dans la direction du gradient de la fonction objectif, tout comme le ferait la descente de gradient, mais en ajoutant la contrainte que les déplacements successifs doivent être conjugués. Les contraintes d'orthogonalité ainsi ajoutées au problème, la minimisation devient beaucoup plus efficace qu'une technique de descente de gradient seule, et converge en moins d'itérations vers le minimum. Toutefois, ce processus de minimisation nécessite que le problème soit bien conditionné et il est en pratique nécessaire de faire appel à un préconditionneur avant de lancer la minimisation. Cette technique d'optimisation non contrainte munie d'un préconditionneur approprié est réputée être très efficace notamment dans les problèmes larges (beaucoup de paramètres) et est une alternative possible à l'algorithme de LEVENBERG-MARQUARDT [Nocedal 1999, Frandsen 1999].

2.2.2.2 Méthode de Gauss-Newton

Comme nous l'avons précisé, l'algorithme de descente de gradient converge très mal près du minimum. La technique d'optimisation non-linéaire de GAUSS-NEWTON est, à l'inverse, une technique particulièrement efficace lorsque la solution initiale est près du minimum de la fonction objectif.

Cette méthode itérative propose un déplacement $\boldsymbol{\delta}_{GN}^{(k)}$ pour chaque itération k en résolvant une approximation locale du problème. GAUSS-NEWTON utilise une approximation linéaire des résidus et, par conséquent, une approximation quadratique de la fonction de coût F. Cette approximation $m_k(\boldsymbol{\delta})$ est définie par :

$$m_k(\boldsymbol{\delta}) = \frac{1}{2}\|\boldsymbol{\Delta}(\mathbf{x}) + \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta}\|^2 \tag{2.29}$$

$$m_k(\boldsymbol{\delta}) = \frac{1}{2}(\boldsymbol{\Delta}(\mathbf{x}) + \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta})^{\top}(\boldsymbol{\Delta}(\mathbf{x}) + \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta}) \tag{2.30}$$

$$m_k(\boldsymbol{\delta}) = \frac{1}{2}\boldsymbol{\Delta}(\mathbf{x})^{\top}\boldsymbol{\Delta}(\mathbf{x}) + \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\Delta}(\mathbf{x})\boldsymbol{\delta}^{\top} + \frac{1}{2}\boldsymbol{\delta}^{\top}{\mathcal{J}_{\Delta}^{(k)}}^{\top}\mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta}. \tag{2.31}$$

Pour chaque itération de l'algorithme de GAUSS-NEWTON, nous cherchons alors le déplacement qui minimise l'approximation courante de la fonction de coût :

$$\boldsymbol{\delta}^{(k)} = \arg\min_{\boldsymbol{\delta}} m_k(\boldsymbol{\delta}). \tag{2.32}$$

Il s'agit ici d'un problème quadratique (donc convexe) et un point stationnaire peut être estimé par l'annulation de la dérivée de l'approximation $m_k(\boldsymbol{\delta})$ par rapport au déplacement $\boldsymbol{\delta}$:

$$\frac{\partial m_k}{\partial \boldsymbol{\delta}}(\boldsymbol{\delta}) = \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\Delta}(\mathbf{x}^{(k)}) + {\mathcal{J}_{\Delta}^{(k)}}^{\top}\mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta}^{(k)} = 0. \tag{2.33}$$

L'équation (2.33) peut être réécrite sous la forme connue sous le nom d'équations normales à l'itération k :

$${\mathcal{J}_{\Delta}^{(k)}}^{\top}\mathcal{J}_{\Delta}^{(k)}\boldsymbol{\delta}^{(k)} = -\mathcal{J}_{\Delta}^{(k)}\boldsymbol{\Delta}(\mathbf{x}^{(k)}). \tag{2.34}$$

Il s'agit ici d'un problème de moindres carrés linéaire pour lequel les techniques de résolution de carrés linéaires décrites dans la section 2.2.1 peuvent être employées (avec $\mathcal{L} = \mathcal{J}_{\Delta}^{(k)}$ et $\mathbf{x} = \boldsymbol{\delta}^{(k)}$). Notons $\boldsymbol{\delta}_{GN}^{(k)}$ le déplacement estimé.

Cette technique possède des propriétés intéressantes du point de vue de la convergence. En effet, si le point de départ $\mathbf{x}^{(0)}$ est suffisamment près du minimum $\mathbf{x}^*$, alors la série $\{\mathbf{x}^{(0)} \dots \mathbf{x}^{(k)}\}_{k\to\infty}$ converge vers $\mathbf{x}^*$ de façon quadratique, et la séquence des normes des gradients $\|\boldsymbol{\nabla}^{(k)} = \mathcal{J}_{\Delta}^{(k)}\boldsymbol{\Delta}(\mathbf{x}^{(k)})\|$ converge aussi quadratiquement vers 0, ce qui rend compte de la stabilité de la convergence [Nocedal 1999]. De plus, toujours dans les conditions où l'on se trouve près du minimum, la méthode de GAUSS-NEWTON fournit une amplitude de déplacement qui est déjà très efficace voire optimale lorsque la fonction de coût est localement lisse. Par la suite, nous verrons que la matrice $\mathcal{B}_{\Delta}^{(k)} = {\mathcal{J}_{\Delta}^{(k)}}^{\top}\mathcal{J}_{\Delta}^{(k)}$ est une approximation de la matrice hessienne $\mathcal{H}_{\Delta}^{(k)}$, particulièrement précise lorsque la fonction de coût est localement lisse.

Lorsque la solution initiale $\mathbf{x}^{(0)}$ se trouve être assez éloignée de l'optimum recherché et que la fonction de coût est rugueuse (fortement non-linéaire, forte courbure), la méthode de GAUSS-NEWTON est peu adaptée car elle converge très lentement. En effet, l'approximation locale de la fonction de coût donne beaucoup d'importance à la courbure de la fonction à la solution courante du problème et la méthode peut alors stagner sur des minima locaux.

Méthode de quasi-Newton. La méthode de NEWTON dans sa version originale nécessite d'estimer les dérivées secondes de la fonction de résidus, mais en pratique, la méthode de GAUSS-NEWTON utilise l'approximation de la hessienne suivante $\mathcal{H}_{\Delta}^{(k)} \approx \mathcal{B}_{\Delta}^{(k)} = {\mathcal{J}_{\Delta}^{(k)}}^{\top} \mathcal{J}_{\Delta}^{(k)}$. Cependant, pour que l'algorithme de GAUSS-NEWTON puisse fonctionner et proposer un déplacement qui diminue la fonction de coût $\boldsymbol{\delta}^{\top} \boldsymbol{\nabla} < 0$, il est nécessaire que cette approximation $\mathcal{B}_{\Delta}^{(k)}$ soit définie positive, ce qui n'est aucunement garanti par la formule précédente. Pour éviter ces difficultés, certaines techniques dites de quasi-Newton ont été proposées (*Broyden [Broyden 1965], (L-)BFGS [Liu 1989], SR1, entre autres*) [Nocedal 1999]. L'idée générale de ces méthodes est de mettre à jour à chaque itération de l'optimisation la matrice Hessienne approximée $\mathcal{B}_{\Delta}^{(k+1)}$ (et son inverse) à partir de l'estimée précédente $\mathcal{B}_{\Delta}^{(k)}$. La fonction de mise à jour diffère suivant les méthodes, tout comme l'estimée initiale $\mathcal{B}_{\Delta}^{(0)}$.

2.2.2.3 Recherche linéaire

Nous venons de présenter les algorithmes les plus fréquemment utilisés pour estimer une direction de déplacement $\boldsymbol{\delta}^{(k)}$ dans l'espace de la paramétrisation locale du modèle. Mais ces algorithmes ne s'attachent pas à calculer la norme (ou l'amplitude) du déplacement à réaliser dans la direction $\boldsymbol{\delta}^{(k)}$. C'est ce que proposent de faire les techniques de recherche linéaire (*line search* en anglais) en tentant de déterminer l'amplitude optimale du déplacement, qui minimise le plus efficacement possible la fonction objectif du problème. Cela permet à l'algorithme de minimisation de converger vers la solution du problème en moins d'itérations. Une illustration d'une étape de la minimisation par recherche linéaire est présentée en figure 2.1.

Pour résoudre ce problème, il existe deux types de recherche linéaire : les méthodes exactes et inexactes.

Méthodes de recherche linéaire exactes. Les techniques de recherche linéaire exacte tentent de résoudre à chaque itération de la minimisation l'équation suivante :

$$\alpha^{(k)\star} = \arg \min_{\alpha^{(k)}>0} F(\mathbf{x}^{(k)} + \alpha^{(k)} \boldsymbol{\delta}^{(k)}). \tag{2.35}$$

Lorsque la fonction objectif est quadratique : $F(\mathbf{x}) = \mathbf{f} + \mathbf{g}^{\top}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\top}\mathcal{H}\mathbf{x}$, la solution exacte de ce problème est la suivante [Nocedal 1999, Frandsen 1999, Higham 2001] :

$$\alpha^{(k)\star} = -\frac{\mathbf{g}^{\top}\boldsymbol{\delta}^{(k)}}{{\boldsymbol{\delta}^{(k)}}^{\top}\mathcal{H}\boldsymbol{\delta}^{(k)}}. \tag{2.36}$$

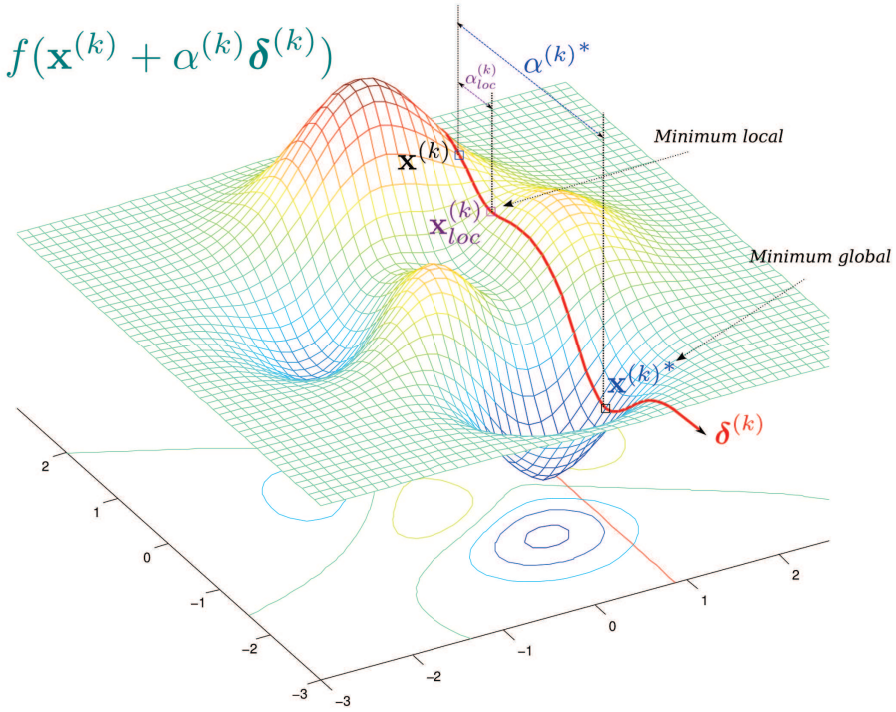


FIGURE 2.1 – Recherche linéaire : estimation de la norme α du déplacement $\boldsymbol{\delta}$. Exemple sur une fonction objectif à deux variables. Deux longueurs sont proposées (α_{loc} et α^*), qui correspondent à deux solutions : $\mathbf{x}_{loc}$ est un minimum local et $\mathbf{x}^*$ est le minimum global dans la direction du déplacement $\boldsymbol{\delta}$.

Malheureusement la majorité des problèmes de vision par ordinateur possède des fonctions objectives $F(\mathbf{x})$ non-linéaires et l'équation (2.35) devient alors difficile à résoudre. Dans ces conditions, il est beaucoup plus intéressant de faire appel aux algorithmes de recherche linéaire inexacte.

Méthodes de recherche linéaire inexactes. De nombreux travaux ont été effectués sur la recherche inexacte de l'amplitude optimale de déplacement pour les problèmes non-linéaires.

Recherche linéaire inexacte par ajustement (*fitting*). Les premières méthodes de recherche linéaire inexactes inventées étaient basées sur une approximation locale de la fonction [Frandsen 1999]. Le principe est d'évaluer la fonction de coût en 3 ou 4 points (estimés par des hypothèses sur l'amplitude du déplacement), puis d'estimer le polynôme de degré 3 ou 4 passant par ces points. Nous obtenons ici une approximation cubique ou quadratique de la fonction objectif du problème. Les racines du polynôme ainsi ajusté forment les solutions inexactes du problème de recherche linéaire. Le choix de la localisation des points est effectué selon plusieurs critères [Frandsen 1999], ce qui augmente le coût calculatoire de la méthode proposée.

Recherche linéaire inexacte par *backtracking*. Les méthodes de recherche linéaire inexactes les plus couramment employées en optimisation non contrainte adoptent un comportement itératif et sont basées sur le principe de *backtracking*. Après avoir réduit un intervalle de recherche, elles initialisent l'amplitude du déplacement sur une première valeur, en prenant par exemple l'unité, puis elles génèrent et évaluent ensuite successivement des hypothèses d'amplitude à l'aide de différents critères (heuristiques), comme par exemple les critères de GOLDSTEIN ou de WOLFE [Nocedal 1999]. Ces méthodes convergent alors plus ou moins rapidement vers une amplitude de déplacement efficace, et tentent ainsi de minimiser d'autant plus la fonction de coût du problème.

De nombreuses techniques de recherche linéaire comme notamment [Al-Baali 1986, Moré 1994, Nocedal 1992] ou [Frandsen 1999], dérivent du principe de *backtracking*. Les auteurs de [Frandsen 1999] et [Al-Baali 1986] décrivent une technique de *line search* itérative (*Soft Line Search*) dans laquelle ils estiment un intervalle efficace ne contenant que des solutions acceptables, c'est-à-dire respectant les critères de WOLFE (voir ci-dessous). La méthode proposée s'avère être très efficace mais aussi très lente étant donnée la discrétisation de l'espace et les multiples appels à la fonction de coût, malgré l'utilisation d'une interpolation polynomiale de la fonction objectif. LIU et NOCEDAL [Liu 1989] font aussi appel aux critères de WOLFE dans le cadre d'une minimisation quasi-Newtonienne limitée (*L-BFGS*).

Certains auteurs emploient la recherche linéaire en conjonction avec la méthode des gradients conjugués. SHANNO [Shanno 1978] a présenté pour la première fois un algorithme d'optimisation non-linéaire par les gradients conjugués avec une méthode de recherche linéaire inexacte, puis HAGER et ZHANG dans [Hager 2006a, Hager 2006b] ont étudié la convergence de la méthode du gradient conjugué munie d'une recherche linéaire inexacte basée sur les conditions (*strong*) de WOLFE. D'autres auteurs [Nocedal 1992] combinent les méthodes duales des régions de confiance et de recherche linéaire. Ils proposent d'employer une recherche linéaire inexact par *backtracking* lorsque l'hypothèse d'amplitude de confiance effectuée par la méthode des régions de confiance est erronée et implique une augmentation de la fonction objectif. Cette méthode possède l'avantage de conserver les propriétés de convergence des méthodes de région de confiance. Les auteurs de [Yuan 2009] ont proposé un algorithme similaire mais utilisent la méthode BFGS dans le cadre de l'optimisation d'un problème pour lequel les équations sont non-linéaires symétriques.

Nous renvoyons le lecteur au chapitre écrit par NOCEDAL et WRIGHT [Nocedal 1999, Chap. 3] consacré aux méthodes classiques de *Line Search* couramment employées en optimisation non-linéaire.

Critères de sélection de l'amplitude de déplacement. Il existe différents critères permettant de caractériser la qualité de l'amplitude du déplacement proposé lors d'une itération de la minimisation. Ces critères ne sont donc employés que pour valider l'efficacité d'une amplitude de déplacement et non pour générer les hypothèses sur la norme. Nous présentons dans cette partie les conditions les plus usités de l'état-de-l'art : les conditions de WOLFE et de GOLDSTEIN.

Conditions de WOLFE. La première condition de WOLFE, aussi nommée critère d'ARMIJO, garantie une décroissance suffisante de l'erreur. Elle s'écrit :

$$F(\mathbf{x}^{(k+1)}) \leq F(\mathbf{x}^{(k)}) + \omega_1 \alpha^{(k)} \boldsymbol{\nabla}^{(k)^\top} \boldsymbol{\delta}^{(k)}. \tag{2.37}$$

Le second critère de WOLFE contraint l'algorithme de recherche linéaire à choisir une norme $\alpha^{(k)}$ minimisant la courbure de la fonction objectif :

$$\boldsymbol{\nabla}^{(k+1)^\top} \boldsymbol{\delta}^{(k)} \leq -\omega_2 \boldsymbol{\nabla}^{(k)^\top} \boldsymbol{\delta}^{(k)}, \tag{2.38}$$

avec $0 < \omega_1 < \omega_2 < 1$. Ces derniers coefficients sont des paramètres que l'utilisateur doit définir et permettent de caractériser l'importance que l'on accorde aux contraintes. En pratique, les valeurs typiques de ces paramètres dans le cadre de méthodes de recherche linéaire par *back-tracking* sont $\omega_1 = 10^{-4}$ et $\omega_2 = 0.9$.

Une illustration graphique des conditions de WOLFE est présenté dans la figure 2.2.

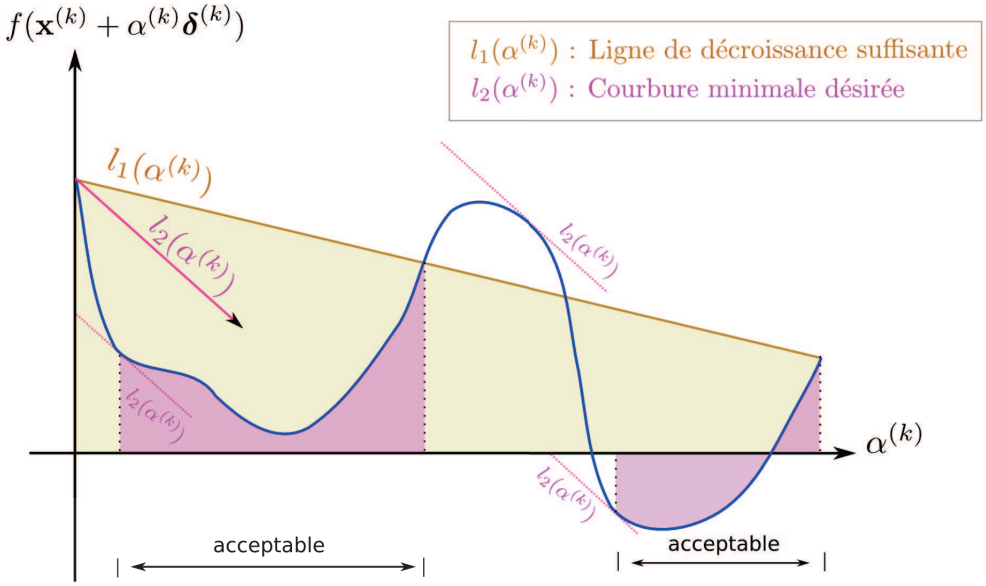


FIGURE 2.2 – Conditions de WOLFE.

Une variante des conditions de WOLFE, connue sous le nom de conditions fortes de WOLFE, apporte une modification sur la deuxième condition et contraint les valeurs absolues des courbures :

$$\left| \boldsymbol{\nabla}^{(k+1)^\top} \boldsymbol{\delta}^{(k)} \right| \leq \omega_2 \left| \boldsymbol{\nabla}^{(k)^\top} \boldsymbol{\delta}^{(k)} \right|. \tag{2.39}$$

Le théorème de ZOUTENDIJK [Nocedal 1999, Chap. 3] montrent que les algorithmes de minimisation de GAUSS-NEWTON intégrant une méthode de recherche linéaire imposant les conditions de WOLFE, ont la garantie de converger vers un point stationnaire ($\lim_{k\to\infty} \left\| \boldsymbol{\nabla}^{(k)} \right\| = 0$).

Conditions de GOLDSTEIN. D'autres critères sur le choix de l'amplitude du déplacement ont été proposés dans la littérature [Nocedal 1999, Chap. 3]. Les conditions de GOLDSTEIN sont assez similaires aux conditions de WOLFE. Leur principe est d'une part de s'assurer que l'amplitude $\alpha^{(k)}$ diminue assez la fonction de coût et d'éviter les déplacement trop petits. Ces conditions peuvent être mises sous la forme d'inégalités :

$$F(\mathbf{x}^{(k)}) + (1-c)\alpha^{(k)}{\boldsymbol{\nabla}^{(k)}}^{\top}\boldsymbol{\delta}^{(k)} < F(\mathbf{x}^{(k+1)}) \leq F(\mathbf{x}^{(k)}) + c\alpha^{(k)}{\boldsymbol{\nabla}^{(k)}}^{\top}\boldsymbol{\delta}^{(k)}. \tag{2.40}$$

avec $0 < c < \frac{1}{2}$. Les critères de GOLDSTEIN plus fréquemment utilisées dans les minimisations de type GAUSS-NEWTON, alors que les conditions de WOLFE sont plus adaptés dans les minimisations quasi-Newtonnienne.

Ces différentes techniques de recherche linéaire de l'état de l'art évaluent la fonction de coût pour chaque hypothèse d'amplitude du déplacement et peuvent ainsi impliquer de nombreuses évaluations sans pour autant garantir une amélioration, un déplacement au plus près de la solution. En pratique, ces méthodes sont coûteuses en temps de calcul et la réduction du nombre d'itérations de la minimisation qu'elles peuvent apporter est généralement compensé par le temps de calcul supplémentaire qu'elles nécessitent. Néanmoins, ces techniques peuvent parfois améliorer la convergence et évitent certains minima locaux de la fonction de coût.

2.2.2.4 Région de confiance

À l'opposé des méthodes de recherche linéaire, on trouve les techniques de région de confiance. Au lieu d'estimer une direction $\boldsymbol{\delta}^{(k)}$ puis l'amplitude $\alpha^{(k)}$ du déplacement pour chaque itération, les méthodes de région de confiance tentent de trouver le meilleur déplacement possible dans un espace réduit dit de confiance.

Ces techniques reposent sur un *a priori*, donné en paramètre à l'algorithme : le rayon de confiance $\Gamma^{(k)}$. Cette limite forme l'hypersphère dans l'espace des paramètres du déplacement au delà de laquelle l'algorithme ne fait plus confiance à l'approximation locale $m_k(\boldsymbol{\delta})$. Ainsi, l'algorithme estime un déplacement $\boldsymbol{\delta}^{(k)}$ (direction et amplitude) répondant au mieux à ce critère courant, tel que $\|\boldsymbol{\delta}^{(k)}\| \leq \Gamma^{(k)}$ et résout le problème sous contraintes suivant :

$$\begin{array}{c} \min_{\boldsymbol{\delta}} m_k(\boldsymbol{\delta}) = \boldsymbol{\Delta}^2(\mathbf{x}) + \boldsymbol{\nabla}^{(k)}\boldsymbol{\delta}^{\top} + \frac{1}{2}\boldsymbol{\delta}^{\top}\mathcal{B}_{\Delta}^{(k)}\boldsymbol{\delta} \\ \text{sous la contrainte} \quad \|\boldsymbol{\delta}\| \leq \Gamma^{(k)} \end{array}. \tag{2.41}$$

Le rayon de confiance est ensuite adaptée par des règles de mise à jour propre à chaque algorithme, et permet ainsi une adaptation rapide de la contrainte suivant le problème (le rayon est diminué lorsque l'on est près d'un minimum et augmenté dans le cas contraire).

Nous décrivons ici les deux principaux algorithmes d'optimisation de moindres carrés par région de confiance : les méthodes *Dog-Leg* et LEVENBERG-MARQUARDT.

Algorithme *Dog-Leg*. L'algorithme *Dog-Leg* [Powell 1970] est un des premiers algorithmes de région de confiance proposé dans la littérature. La résolution du problème contraint (2.41) est

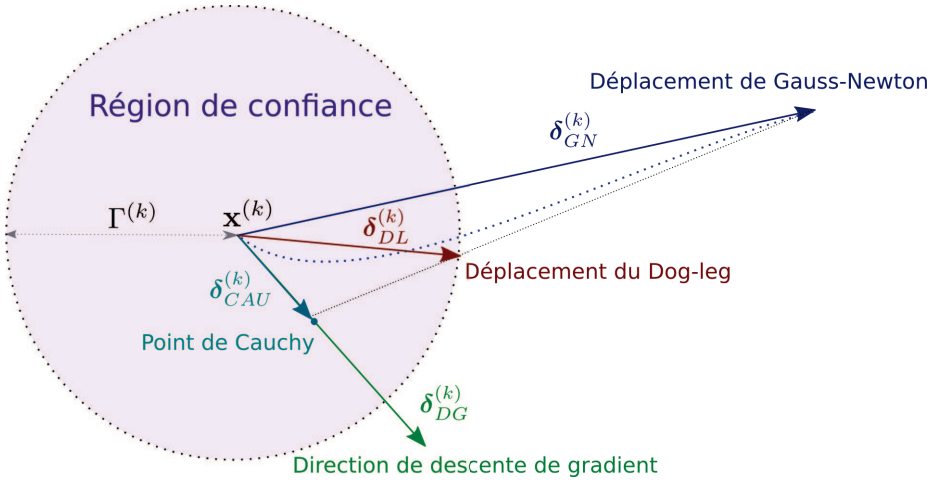


FIGURE 2.3 – Illustration de l'algorithme *Dog-Leg* et des différents déplacements mis en jeu à l'itération k.

réalisée par une interpolation entre les deux méthodes d'optimisation décrites précédemment : les méthodes de GAUSS-NEWTON et de descente de gradient.

Pour cela, l'algorithme estime à chaque itération le déplacement optimal du sous problème non contraint (sans la contrainte sur l'amplitude de déplacement) selon les méthodes de GAUSS-NEWTON et de descente de gradient et calcule ensuite le point de CAUCHY, c'est-à-dire le déplacement dont l'amplitude est optimale le long de la direction de descente de gradient. Une illustration est proposée en figure 2.3. Le point de CAUCHY correspond à :

$$\boldsymbol{\delta}_{CAU}^{(k)} = \alpha_{DG}^{\star} \boldsymbol{\nabla}^{(k)}, \quad \text{avec } \alpha_{DG}^{\star} = \frac{{\boldsymbol{\nabla}^{(k)}}^{\top} \boldsymbol{\nabla}^{(k)}}{{\boldsymbol{\nabla}^{(k)}}^{\top} \mathcal{B}_{\Delta}^{(k)} \boldsymbol{\nabla}^{(k)}}. \tag{2.42}$$

L'algorithme *Dog-Leg* opère ensuite une interpolation entre ces deux déplacements en définissant le polygone $\boldsymbol{\delta}_{DL}^{(k)}(\tau^{(k)})$ paramétrée par $\tau^{(k)}$ qui prend le déplacement le plus long au regard de la contrainte du rayon de confiance $\Gamma^{(k)}$. Ce polygone définit la trajectoire *Dog-Leg* et est tel que :

$$\boldsymbol{\delta}_{DL}^{(k)}(\tau^{(k)}) = \begin{cases} \tau^{(k)} \boldsymbol{\delta}_{CAU}^{(k)}, & 0 \leq \tau^{(k)} \leq 1, \\ \boldsymbol{\delta}_{CAU}^{(k)} + (\tau^{(k)} - 1)(\boldsymbol{\delta}_{GN}^{(k)} - \boldsymbol{\delta}_{CAU}^{(k)}), & 1 < \tau^{(k)} \leq 2. \end{cases} \tag{2.43}$$

L'algorithme résout ensuite le problème non-linéaire en τ : $\boldsymbol{\delta}_{DL}^{(k)}(\tau^{(k)}) = \Gamma^{(k)}$.

La méthode combine ainsi les avantages des algorithmes de GAUSS-NEWTON et de descente de gradient. Grâce à un mécanisme de mise à jour du rayon de confiance, cette méthode privilégie l'une ou l'autre méthode et converge ainsi rapidement vers un minimum de la fonction objectif. Dans cette thèse nous avons implémenté l'algorithme *Dog-Leg* proposé par LOURAKIS *et al.* [Lourakis 2005] et nous utilisons la valeur initiale et les règles de mise à jour du rayon de confiance $\Gamma^{(k)}$ décrites dans l'article.

Cette technique est adaptée aux problèmes bien conditionnés mais en pratique les problèmes sont souvent mal conditionnés et la méthode peut présenter des problèmes de rapidité voire de qualité de convergence. Pour palier ces difficultés, nous pouvons utiliser une région de confiance ellipsoïdale au lieu de sphérique. Une technique utilisant ce procédé est la méthode de LEVENBERG-MARQUARDT.

Algorithme de LEVENBERG-MARQUARDT. L'algorithme de LEVENBERG-MARQUARDT [Levenberg 1944, Marquardt 1963] est une méthode pratique de résolution du problème contraint des régions de confiances. Cette méthode transforme le problème contraint en un problème non contraint en introduisant un terme d'amortissement (*damping*) ou de régularisation λ (voir la section 2.2.1.2). En effet, il peut être montré que la solution $\boldsymbol{\delta}^*$ du problème contraint (2.41) est aussi la solution du problème non contraint suivant :

$$(\mathcal{B}_\Delta + \lambda^2 \mathcal{D})\boldsymbol{\delta}^* = -\boldsymbol{\nabla}. \tag{2.44}$$

pour tout $\lambda \geq 0$ et où $\mathcal{B}_\Delta$ est une matrice semi-définie positive approximant la matrice hessienne. $\mathcal{D}$ est une matrice définie positive dite de régularisation, qui peut être simplement la matrice identité $\mathcal{D} = \mathcal{I}$, ou la diagonale de la matrice $\mathcal{B}_\Delta$.

Le principal avantage de l'algorithme de LEVENBERG-MARQUARDT réside dans sa formulation en un problème non contraint. Cette formulation permet à l'algorithme d'adapter simplement en interne son paramètre d'amortissement et ainsi passer d'un comportement de Descente de Gradient lorsque celui-ci se trouve être loin de l'optimum à un comportement de GAUSS-NEWTON lorsqu'il est près de l'optimum de la fonction de coût et ainsi bénéficier d'une convergence quadratique. À la différence de la régularisation des systèmes d'équations linéaires (section 2.2.1.2) dédiée à la résolution des problèmes mal posés, dans l'algorithme de LEVENBERG-MARQUARDT la valeur du paramètre λ va contrôler l'importance que l'on accorde aux composantes principales (dérivées secondes selon chaque paramètres de $\mathbf{x}$) par rapport aux composantes secondaires (dérivées inter-paramètres) dans le choix du déplacement à effectuer.

Paramètre d'amortissement. Contrairement à l'algorithme *Dog-Leg* qui définit à chaque itération de l'optimisation un rayon de confiance $\Gamma^{(k)}$, dans la méthode de LEVENBERG-MARQUARDT le paramètre d'amortissement influence simultanément la direction et l'amplitude du déplacement. Cette action double implique que le choix de la valeur initiale $\lambda^{(0)}$ est plus complexe que dans la méthode du *Dog-Leg*. La convergence de l'algorithme de LEVENBERG-MARQUARDT peut être démontrée sous certaines restrictions sur la valeur initiale du paramètre d'amortissement. Cependant, ces restrictions ne sont généralement pas employées puisque de meilleurs performances (en termes de vitesse de convergence) sont obtenues avec une valeur initiale choisie de manière empirique. En pratique, nous fixons la valeur initiale $\lambda^{(0)}$ à 10^{-3}.

Le paramètre d'amortissement est ensuite mis-à-jour à chaque itération de la minimisation. Pour réaliser cela, plusieurs heuristiques de mise à jour ont été proposées dans la littérature [Nocedal 1999, Nielsen 1999, Fasel 2008]. Dans cette thèse nous avons choisi d'utiliser les rè-

gles de mise à jour décrites dans [Hartley 2003b] :

$$\lambda^{(k+1)} = \begin{cases} \lambda^{(k)}/\nu_1 & \text{si } F(\mathbf{x}^{(k)}) < F(\mathbf{x}^{(k-1)}) \\ \lambda^{(k)} * \nu_2 & \text{si } F(\mathbf{x}^{(k)}) > F(\mathbf{x}^{(k-1)}) \end{cases} . \tag{2.45}$$

avec les constantes $\nu_1 = 10, \nu_2 = 10$.

Un deuxième intérêt de cette formulation est la possibilité de définir simplement une région de confiance ellipsoïdale (et non plus sphérique) en remplaçant la matrice $\mathcal{D}$ de l'équation (2.44) par une matrice diagonale composée par exemple des éléments diagonaux de l'approximation de la matrice hessienne $\mathcal{D} = \text{diag}({\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta)$. Le déplacement ainsi estimé est alors confiné sur une ellipse définie par cette matrice et cela correspond alors à une optimisation sous la contrainte $\|\mathcal{D}\boldsymbol{\delta}\| \leq \Gamma$.

2.2.3 Moindres carrés non-linéaires multiobjectif

Principe général. Dans cette thèse nous nous intéressons à la fusion d'un ensemble d'observations issues de plusieurs (N_K) capteurs. Chaque capteur délivrant un ensemble d'observations qui lui est propre, nous pouvons construire pour chaque capteur un terme d'erreur, noté $F_k(\mathbf{x})$, correspondant aux résidus entre les observations du capteur k et les prédictions du modèle. Nous cherchons dès lors les paramètres optimaux $\mathbf{x}^*$ du modèle expliquant au mieux l'ensemble de toutes les observations des capteurs. Une manière de faire est d'écrire ce problème sous la forme d'un problème d'optimisation multiobjectif, aussi connu sous le nom d'*inversion jointe* , où l'on s'attachera à trouver les meilleurs paramètres $\mathbf{x}^*$ minimisant simultanément tous les termes d'erreur, tous les objectifs :

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathfrak{M}} \{F_1(\mathbf{x}), F_2(\mathbf{x}), \ldots, F_{N_K}(\mathbf{x})\}. \tag{2.46}$$

Ce type de problème est difficile à résoudre puisque d'une part, l'espace de recherche des paramètres du modèle peut être très important, à l'image des problèmes d'optimisation mono-objectif, et d'autre part ces derniers possèdent généralement une infinité de solutions : la frontière de PARETO.

Frontière de Pareto. La frontière de Pareto représente l'ensemble des solutions du problème (2.46) pour lesquelles tous les objectifs sont satisfaits. Une illustration des solutions $\mathbf{x}$ pour un problème à deux objectifs est donnée en figure 2.4. L'espace des solutions possibles sur ce problème bi-objectif d'exemple est représenté par la zone violette et la ligne de Pareto est représentée par la ligne bleue.

Une solution $\mathbf{x}^\star$ est dite *Pareto optimale* s'il n'existe pas de vecteur $\mathbf{y}$ pour lequel l'ensemble des objectifs de $\mathbf{x}$ sont inférieurs ou égaux aux objectifs de $\mathbf{y}$: $F_k(\mathbf{x}) \leq F_k(\mathbf{y}), \forall k \in [1, N_K]$, et qu'il existe au moins un objectif strictement inférieur : $\exists k \in [1, N_K] | F_k(\mathbf{x}) < F_k(\mathbf{y})$. Les solutions Pareto optimales sont notées avec une étoile (et par un astérix) sur la figure 2.4.

Pour trouver la meilleure solution (notée $\mathbf{x}^*$) parmi cette frontière, meilleure au sens qu'elle minimise au mieux tous les objectifs du problème, une manière de faire est de prendre la solution

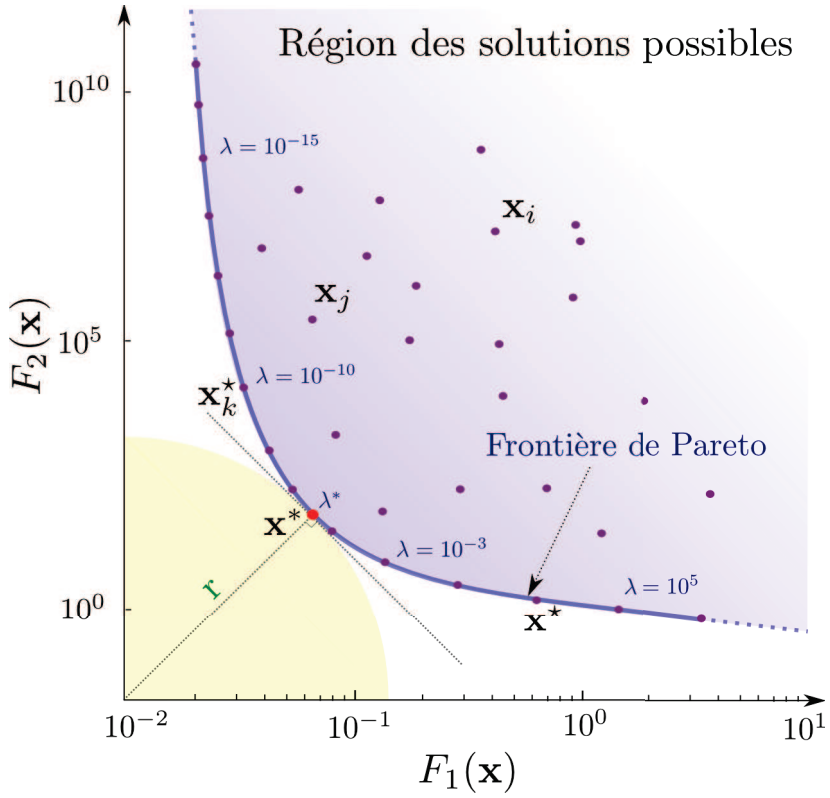


FIGURE 2.4 – Solutions possibles et frontière de Pareto pour un problème simple à deux objectifs $\{F_1, F_2\}$.

$\mathbf{x}^*$ qui est la plus près de l'origine (voir figure 2.4). Cette technique ne fonctionne en revanche que lorsque la frontière de PARETO est convexe. Il est de plus possible de ne générer que les solutions présentes sur la frontière de PARETO, et ainsi limiter l'espace de recherche à parcourir. Pour cela, il suffit de regrouper tous les objectifs en une seule fonction de coût composée, dont chaque objectif est pondéré par un coefficient à déterminer, noté λ_l (voir la section suivante).

2.2.3.1 Fonction de coût à objectifs pondérés

Le principe de l'*Aggregate Objective Function* est de sommer et pondérer tous les objectifs du problème (2.46) et ainsi construire une seule et unique fonction de coût $F(\mathbf{x}; \{\ldots, \lambda_k, \ldots\})$ telle que :

$$F(\mathbf{x}; \{\lambda_1, \ldots, \lambda_{N_K}\}) = \sum_{k=1}^{N_K} \lambda_k^2 F_k(\mathbf{x}), \tag{2.47}$$

où λ_k^2 est la pondération de l'objectif F_k. Nous verrons par la suite que le paramètre est élevé au carré afin de correspondre à l'inverse de l'écart-type de l'objectif dans les problèmes de moindres carrés. Ces paramètres de pondération représentent la confiance ou l'importance que l'on accorde à tel ou tel objectif, et caractérisent le modèle que l'on utilise pour modéliser chaque capteur.

Le problème d'optimisation multiobjectif est alors le suivant :

$$\min_{\mathbf{x}} F(\mathbf{x}; \{\lambda_1, \ldots, \lambda_{N_K}\}). \tag{2.48}$$

L'approche classique est séquentielle : on recherche en premier lieu les meilleurs paramètres de pondération λ_k, puis on applique ensuite directement les techniques de minimisation non-linéaires à un seul objectif présentées au début du chapitre. Le principal avantage de cette formulation est que les solutions générées par la minimisation de cette unique fonction selon différentes valeurs de pondérations sont positionnées directement sur la frontière de Pareto. Tout le problème revient donc à savoir quelles sont les meilleures valeurs des pondérations λ_k.

Dans cette thèse nous allons nous restreindre à deux objectifs ($N_K = 2$) et nous présenterons donc les techniques de sélection des paramètres de pondération adaptées à ce type de problème. Pour simplifier les notations, nous noterons la fonction bi-objectif pondérée, paramétrant la frontière de Pareto (voir figure 2.4), de la manière suivante :

$$F(\mathbf{x}; \{1, \lambda\}) = F_1(\mathbf{x}) + \lambda^2 F_2(\mathbf{x}). \tag{2.49}$$

Sélection des paramètres de pondération. Il existe différents procédés pour sélectionner les paramètres de pondération : par étalonnage, essai-erreur, critères ou apprentissage. Lorsque l'on peut modéliser de manière précise la fonction de mesure des capteurs par étalonnage (biais et écart-type), on peut montrer grâce à l'équation (2.11) que les pondérations recherchées (dans les estimateurs au maximum de vraisemblance) correspondent à l'inverse des écarts-types des capteurs $\lambda_k = \frac{1}{\sqrt{2}\sigma_k}$ (ou au ratio dans le cas bi-objectif $\lambda = \frac{\sigma_1}{\sigma_2}$). En ce sens, il n'est pas nécessaire de faire appel aux heuristiques de sélection des pondérations sauf quand notre confiance en ces modèles est faible. Dans le cas où les écarts-types des capteurs sont connus, le problème (2.49) devient un problème d'optimisation mono-objectif et les techniques d'optimisation non-linéaires précédemment décrites peuvent être employées.

Lorsqu'il n'est pas possible de modéliser précisément les processus d'observation des capteurs ou lorsque l'on souhaite insérer un *a priori* sous la forme d'un terme d'erreur générique (moindre carré), le paramètre de pondération doit être sélectionné automatiquement. Tout l'enjeu ici est de trouver le paramètre de pondération qui fournit le meilleur compromis entre les deux objectifs. Nous présentons dans la section 2.4 quelques unes des techniques de sélection automatique du paramètre de pondération, à l'origine dédiées au problème de sélection de complexité.

2.3 Estimation robuste

Nous venons de présenter des méthodes d'estimation des paramètres du modèle lorsque les mesures sont considérées comme correctes (*inliers* en anglais), c'est-à-dire qu'elles suivent une distribution gaussienne centrée. En pratique certaines mesures s'éloignent fortement de cette loi de probabilité et nous parlerons alors de données erronées ou aberrantes (*outliers* en anglais). Ces données ne doivent pas être prises en compte dans les données d'entrée des algorithmes d'estimation des paramètres que nous venons de voir car elles diminueraient fortement la qualité de l'estimation. Pour résoudre ce problème, différentes approches ont été proposées.

2.3.1 RANSAC

La technique RANSAC (*RANdom SAmple Consensus*) proposée par FISCHLER et BOLLES [Fischler 1981] est une technique d'estimation robuste dont le principe est de sélectionner un sous-ensemble minimal de N_o^{IN} mesures correctes $\hat{\mathbf{z}}^{IN}$, parmi les N_o observations du problème, qui offre la meilleure estimation des paramètres du modèle $\hat{\mathbf{x}}$. Pour cela, RANSAC réalise un tirage aléatoire des mesures et sélectionne celles qui possèdent une erreur inférieure à un seuil ξ. Sont considérées comme *inliers* les observations $\hat{z}_i$ telles que $\|\hat{z}_i - h_i(\hat{\mathbf{x}})\| < \xi$. Plus précisément, RANSAC se compose des étapes suivantes :

- **Tirage.** On effectue un tirage aléatoire de N_o^{IN} mesures $\hat{\mathbf{z}}^{IN}$, parmi l'ensemble des N_o observations du problème.
- **Résolution.** On résout ensuite l'équation décrivant le problème étudié à l'aide d'une des techniques d'optimisation par les moindres carrés pour obtenir une solution $\hat{\mathbf{x}}$ à partir des données sélectionnées $\hat{\mathbf{z}}^{IN}$.
- **Mesure de qualité.** On mesure la qualité de la solution $\hat{\mathbf{x}}$ en comptant le nombre d'observations respectant le critère $\|\hat{z}_i - h_i(\hat{\mathbf{x}})\| < \xi$.
- **Sélection du meilleur jeu de données.** On choisit enfin l'échantillon $\hat{\mathbf{z}}^{IN}$ (et la solution $\hat{\mathbf{x}}$) ayant le plus grand nombre de mesures correctes et on ré-estime les paramètres du modèle à partir de toutes les mesures correctes (*inliers*).

Le nombre de tirages aléatoires à effectuer dépend d'un *a priori* sur le ratio entre le nombre de mesures aberrantes et le nombre de mesures correctes et de la probabilité de trouver la solution optimale désirée [Fischler 1981]. une version adaptative du RANSAC sélectionne dynamiquement le nombre de tirages à réaliser en calculant à chaque itération le pourcentage d'*outliers* parmi les données [Hartley 2003b]. Cette technique permet donc de ne sélectionner que les observations correctes et élimine les mesures aberrantes du problème.

Notons qu'il existe des variantes de la méthode RANSAC (PROSAC, MLESAC, ARRSAC, etc.), dont une évaluation est proposée dans [Raguram 2008].

2.3.2 M-estimateurs

Une autre technique permettant de résoudre le problème d'estimation robuste est le M-estimateur. Le principe des M-estimateurs est de pondérer les observations suivant leur mesure de qualité, c'est-à-dire suivant leur terme d'erreur $\Delta_i(\mathbf{x}) = \hat{z}_i - h_i(\mathbf{x})$. En effet, nous pouvons remarquer que dans les problèmes de moindres carrés la contribution de chacun des résidus $\Delta_i(\mathbf{x})$ est quadratique et par conséquent, plus une mesure est aberrante et plus son influence sur la fonction de coût sera importante. Pour éviter cela, les M-estimateurs remplacent le carré dans la fonction de coût :

$$F(\mathbf{x}) = \sum_i \rho(\|\Delta_i(\mathbf{x})\|). \tag{2.50}$$

Plusieurs M-estimateurs ont été proposés dans la littérature [Huber 1981] et l'un des plus employés dans la communauté de vision par ordinateur est celui ce TUKEY [Beaton 1974]. Cet M-estimateur a la propriété intéressante de réduire rapidement les fort résidus (données aberrantes).

Il est définit par la fonction suivante :

$$\rho_{Tukey}(x) = \begin{cases} \frac{\sigma^2}{2}\left(1-\left(1-\left(\frac{x}{c}\right)^2\right)^3\right) & \text{si} |x| \leq \sigma \\ \frac{c^2}{6} & \text{sinon} \end{cases}, \tag{2.51}$$

avec $c = 4.6851\sigma$. Les M-estimateurs nécessitent de fixer un seuil, σ, à partir duquel les points sont considérés aberrants. Ce seuil est en pratique fixé empiriquement mais il peut être automatiquement estimé à partir des résidus mesurés, par exemple avec la méthode MAD (*Median Absolute Deviation*) de MALIS et MARCHAND [Malis 2006].

Dans cette thèse nous avons choisi d'utiliser principalement la technique RANSAC lors de nos optimisations robustes. En effet, RANSAC permet de traiter les configurations dans lesquelles le taux de contamination des données est important, ce qui correspond au contexte de cette thèse.

2.4 Sélection de complexité

Dans cette section nous introduisons brièvement le problème de la sélection de complexité ainsi que quelques unes des techniques employées pour résoudre ces problèmes difficiles.

Le principe de la sélection de modèle est de déterminer le meilleur modèle statistique à utiliser parmi un ensemble de modèles potentiels à partir des données. La sélection de modèle est une tâche délicate car la plupart du temps celle-ci est combinée à l'estimation des paramètres du modèle à partir des mêmes données. Dans le contexte de l'ajustement d'une courbe à partir des observations, les modèles complexes (possédant un grand nombre de paramètres) s'adapteront mieux aux mesures mais introduiront des paramètres supplémentaires non significatifs et inutiles. L'objectif est alors d'établir un compromis à l'aide de critères (*Bayesian Information Criterion*, validation croisée, *Minimum Description Length*, ...) entre la qualité de l'ajustement du modèle sur les données et sa complexité, et entre biais et variance.

Les techniques de sélection de complexité sont souvent utilisées dans les problèmes de régularisation 2.2.1.2 afin d'éviter le surapprentissage (*overfitting*) sur des données bruitées. Dans le contexte de cette thèse, nous utiliserons les techniques de sélection de modèle afin de sélectionner un paramètre de pondération (voir section 2.2.3). Il ne s'agira donc pas d'un problème classique de sélection de modèle puisque le nombre de paramètre de notre modèle restera constant. Nous présentons ici trois critères de sélection de modèle : la validation croisée et les critères *L-curve* et *L-Tangent Norm*.

2.4.1 Sélection par validation croisée

La sélection de modèle peut se faire à l'aide d'une technique issue de la théorie de l'apprentissage : la validation croisée. La validation croisée est un outil d'apprentissage très utilisé qui propose un critère mesurant la qualité de prédiction d'un modèle, basé sur les considérations statistiques suivantes : un modèle adéquat doit prédire avec précision les données manquantes.

Plus précisément, si l'on extrait quelques observations du vecteur d'observations $\hat{\mathbf{z}}$ et réalisons l'apprentissage du modèle (dans notre cas, une optimisation au sens des moindres carrés) sur les données restantes, alors la solution obtenue doit expliquer aussi ces mesures absentes. Le principe de la validation croisée générale (GCV) est alors de séparer les observations en deux ensembles : un ensemble d'apprentissage et un ensemble de test. Il existe plusieurs variantes de validation croisée qui diffèrent principalement dans leur gestion de ces deux ensembles d'apprentissage et de test. Lorsque la quantité des mesures est limitée, ce qui est généralement vrai dans nos applications, il est possible d'utiliser un découpage des ensembles avec recyclage. L'idée est ici de réutiliser les données de test pour l'apprentissage et *vice versa*, de manière cyclique, par exemple en ne prenant qu'une seule observation de test comme le propose la variante *leave-one-out* [Wahbaa 1975].

GOLUB *et al.* ont proposé dans [Golub 1979] une version généralisée de la validation croisée avec laquelle ils proposent d'estimer un paramètre de régularisation. Dans ce problème de régularisation de TIKHONOV (moindres carrés linéaires, voir section 2.2.1.2), la version généralisée de la validation croisée détermine un score, noté $C_{CV}(\lambda)$, pour chaque valeur de régularisation λ. Ce score représente la moyenne des différences entre les prédictions du modèle appris sur une partie des observations (jeu d'apprentissage), et les observations du jeu de test. Dans la variante *leave-one-out* de la validation croisée, le jeu de test est constitué d'une seule observation et le jeu est recyclé avec l'ensemble d'apprentissage. Cette variante est adaptée pour les contextes où le nombre d'observation est réduit. Le score de validation croisée *leave-one-out*, noté $C_{CV_{loo}}$,représente la moyenne des différences entre l'observation $\hat{z}_i$ et la prédiction prévue par les paramètres $\mathbf{x}_{\lambda}^{[\hat{z}_i]}$ du modèles estimés sans l'observation $\hat{z}_i$. Le calcul de ce critère est présenté dans l'algorithme 2 et un exemple de courbe typique est proposé dans la figure 2.5.

Le paramètre de régularisation est alors celui dont l'erreur de prédiction est minimale :

$$\lambda^{*}_{CVloo} = \arg\min_{\lambda} CV_{loo}(\lambda). \tag{2.52}$$

Les auteurs de [Wahbaa 1975, Golub 1979] proposent une méthode d'estimation rapide du score de validation croisée *leave-one-out* à l'aide d'une formule non itérative qui approxime la somme des erreurs de prédictions. Les auteurs de [Farenzena 2008] utiliserons cette version approximée non itérative dans le cadre d'un problème de moindre carrés non-linéaires. Dans cette thèse nous nous sommes restreint à la version générale *leave-one-out* et non approximée du score.

Algorithme 2 : `CrossValidationModelSelection` : Algorithme de sélection de modèle par validation croisée *leave-one-out* généralisée

- Données : Les observations $\hat{\mathbf{z}}$, un ensemble L de modèles : $L = \{\dots, \lambda, \dots\}$
- Entrées : Les paramètres du modèle $\mathbf{x}$
- Sorties : Le modèle sélectionné, représenté par λ^*

pour tous les $\lambda \in L$ **faire**
  $C_{CV_{loo}}(\lambda) \leftarrow 0$;
  pour tous les $\hat{z}_i \in \hat{\mathbf{z}}$ **faire**
    `// Estimation de la solution` $\mathbf{x}_\lambda^{[\hat{z}_i]}$ `pour le modèle` λ `sans la donnée` $\hat{z}_i$
    $\mathbf{x}_\lambda^{[\hat{z}_i]} \leftarrow \arg\min_{\mathbf{x}} F_{[\hat{z}_i];\lambda}(\mathbf{x})$, où $F_{[\hat{z}_i];\lambda}$ est la fonction objective sans l'erreur observée sur la donnée $\hat{z}_i$;
    `// Calcul de l'erreur de prédiction sur la donnée extraite`
    $C_{CV_{loo}}(\lambda) \leftarrow C_{CV_{loo}}(\lambda) + \frac{1}{N_z}\|\hat{z}_i - h_i(\mathbf{x}_\lambda^{[\hat{z}_i]})\|_2^2$

`// Estimation du meilleur modèle minimisant les erreurs de prédiction`
$\lambda^*_{CV_{loo}} \leftarrow \arg\min_{\lambda} C_{CV_{loo}}(\lambda)$

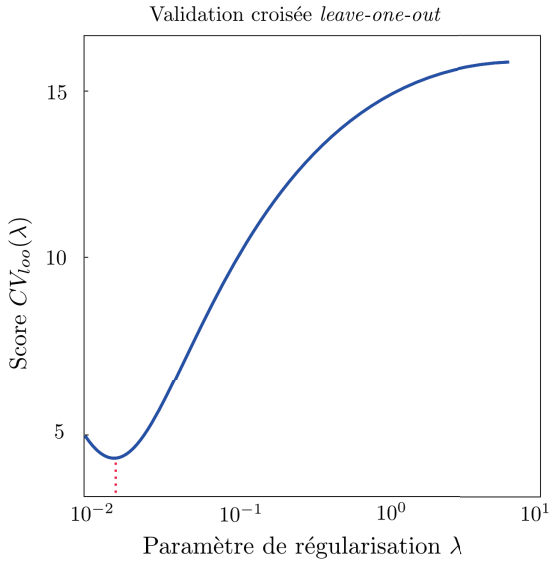


FIGURE 2.5 – Exemple de score de validation croisée *leave-one-out* sur un problème de régularisation.

2.4.2 Sélection par critère *L-curve* et variantes

Plus récemment, des heuristiques ont été proposés pour résoudre le problème de la sélection automatique du paramètre de régularisation dans les problèmes mal posés [Hansen 2001, Regińska 2002, Krawczy-Stando 2007, Brunet 2008, Bazán 2008, Bazán 2009]. Le principe général de ces différentes heuristiques est de calculer un critère sur la relation (*trade-off*) entre les deux objectifs (norme des résidus, norme de la solution) et de choisir le meilleur critère parmi l'ensemble des paramètres régularisation proposés. L'algorithme générique de sélection par critère (ou score) est proposé dans 3.

Algorithme 3 : `CriterionBasedModelSelection` : Algorithme de sélection de modèle par critère .

- Données : Les observations $\hat{\mathbf{z}}$, un ensemble L de modèles : $L = \{\ldots, \lambda, \ldots\}$
- Entrées : Les paramètres du modèle $\mathbf{x}$
- Sorties : Le meilleur modèle, représenté par λ^*

pour tous les $\lambda \in L$ **faire**
  `// Estimation de la solution` $\mathbf{x}_\lambda^* = \mathbf{x}^*(\lambda)$ `pour le modèle` λ
  $\mathbf{x}_\lambda^* \leftarrow \arg\min_{\mathbf{x}} F(\mathbf{x}); \lambda;$
  `// Calcul du score du modèle courant`
  $C(\lambda) \leftarrow$ `ScoreCritère`$(F_1(\mathbf{x}_\lambda^*), F_2(\mathbf{x}_\lambda^*))$;
`// Estimation du meilleur modèle`
$\lambda^* \leftarrow \arg\min_{\lambda} C(\lambda)$;

Nous détaillons dans la suite les fonctions de calcul du score ($ScoreCritere$) pour les critères L-Curve et LTN.

2.4.2.1 *L-Curve*

Le critère *L-Curve* a été initialement développé dans le contexte de la régularisation de TIKHONOV [Hansen 2001]. Le principe de cette technique est de sélectionner le meilleur compromis entre minimiser la norme des résidus (c'est-à-dire $\|\mathcal{H}\mathbf{x}(\lambda) - \hat{\mathbf{z}}\|_2^2$) et minimiser la valeur de la pénalité (c'est-à-dire la norme de la solution pour la régularisation de TIKHONOV $\|\mathcal{D}\mathbf{x}(\lambda)\|_2^2$). Ces deux quantités sont estimées pour différents modèles de régularisation (c'est-à-dire plusieurs valeurs de λ) et sont tracées sous une échelle *log-log*. Dans la plupart des problèmes de régularisation de problèmes mal posés, la courbe ainsi tracée possède une forme typique de L, comme le montre la figure 2.6. Le modèle (c'est-à-dire la valeur du terme de régularisation λ) situé sur le coin du L est alors choisi comme l'optimum et correspond au meilleur compromis entre les deux objectifs. En effet, ce point singulier sépare la partie verticale de la partie horizontale où la solution est dominée par le premier ou le second objectif. La courbe de compromis de la figure correspond à un problème inverse de synthèse sur les équations de la chaleur [Carasso 1982], avec 256

variables et un bruit additionnel de 10^{-3} sur les observations (implémentation de HANSEN [2]).

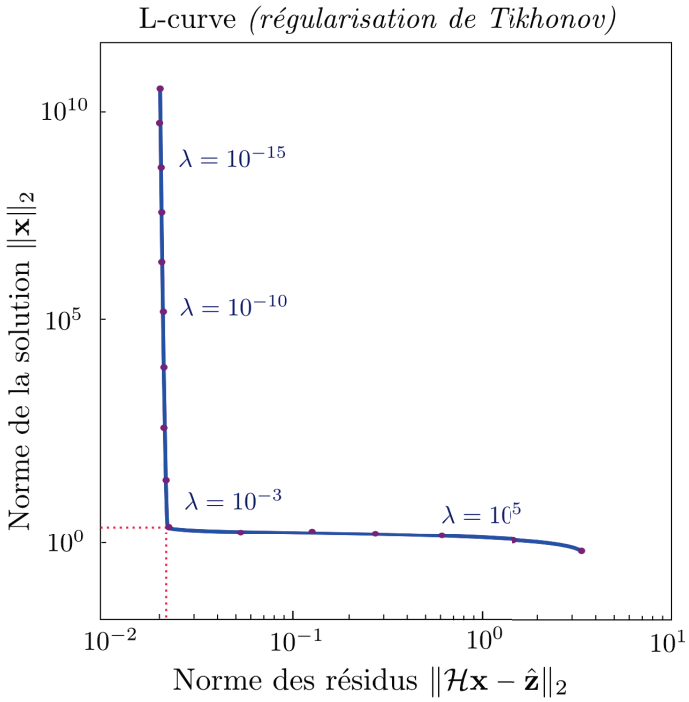


FIGURE 2.6 – *L-Curve* : courbe du compromis entre norme de résidus et norme de la solution sur un problème de régularisation de TIKHONOV.

HANSEN [Hansen 2001] a proposé une technique pour localiser le coin de la courbe à l'aide de l'équation de courbure $\kappa(\lambda)$ suivante :

$$\kappa(\lambda) = 2\frac{\hat{\rho}'\hat{\nu}'' - \hat{\rho}''\hat{\nu}'}{(\hat{\rho}'^2 + \hat{\nu}'^2)^{\frac{3}{2}}}, \tag{2.53}$$

où $\hat{\rho} = \log(\|\mathcal{H}\mathbf{x}(\lambda) - \hat{\mathbf{z}}\|^2)$, $\hat{\nu} = \log(\|\mathcal{D}\mathbf{x}(\lambda)\|^2)$ et $'$ et $''$ désignent les dérivées premières et secondes par rapport au paramètre λ. Le paramètre de régularisation est alors choisi lorsque la courbure est maximale :

$$\lambda^*_{L-Curve} = \arg\min_{\lambda} C_{L-Curve} \text{ avec } C_{L-Curve} = -\kappa(\lambda). \tag{2.54}$$

Cette méthode possède cependant quelques inconvénients, notamment dans les cas où la forme de la courbe n'est pas convexe, lorsqu'elle possède de multiples coins ou lorsque le coin est peu prononcé, lisse. De plus, cette méthode est coûteuse en temps de calcul et nécessite le calcul des dérivées secondes, ce qui la rend sensible aux instabilités numériques. Il existe d'autres techniques pour localiser le coin, voir notamment les travaux de REGIŃSKA [Regińska 2002] et plus récemment de BAZÁN *et al.* [Bazán 2008, Bazán 2009].

2. http://www2.imm.dtu.dk/~pch/Regutools/index.html

2.4.2.2 *L-Tangent Norm* (LTN)

BRUNET *et al.* proposent dans [Brunet 2008] un critère empirique différent du critère *L-Curve*, basé sur la norme de la tangente de la courbe, et non la courbure de celle-ci. Au lieu de rechercher le coin de la courbe du *trade-off*, le critère LTN propose de rechercher le paramètre λ pour lequel une petite variation de sa valeur génère le plus faible impact sur le compromis des deux objectifs. La solution optimale au sens du critère LTN est

$$\lambda^*_{LTN} = \arg\min_{\lambda} C_{LTN}(\lambda) \;\text{ avec }\; C_{LTN}(\lambda) = \|(\bar{\rho}', \bar{\nu}')\|_2^2 \,, \tag{2.55}$$

où $\bar{\rho}$ et $\bar{\nu}$ sont les deux erreurs normalisées (voir [Brunet 2008]). Les auteurs montrent que cette variante de *L-Curve* s'avère être plus rapide à estimer le paramètre de régularisation qu'avec une méthode de type validation croisée. Elle est aussi plus stable que la technique de *L-Curve* originelle puisqu'elle ne nécessite pas le calcul de dérivées secondes.

2.5 Conclusion

Nous venons de présenter les différentes techniques employées afin de résoudre le problème de l'estimation de paramètres, et plus particulièrement de la résolution des problèmes inverses linéaires et non-linéaires, mono et multiobjectif. Dans la suite de ce mémoire nous verrons comment et pourquoi ces techniques d'optimisation sont fréquemment usitées dans de nombreux problèmes de vision par ordinateur, et notamment dans l'ajustement de faisceaux.

La sélection de complexité d'un modèle est aussi un domaine de recherche actif et nous verrons comment nous employons ces méthodes dans le contexte de la fusion de données par ajustement de faisceaux contraint.

CHAPITRE 3

Localisation et reconstruction 3D par vision

Nous présentons dans ce chapitre des algorithmes de localisation et reconstruction 3D par vision, et plus particulièrement les techniques de structure à partir du mouvement (Structure-from-Motion en anglais). Après avoir introduit les algorithmes de base en vision par ordinateur (détection, description et appariement de points d'intérêt, triangulation de point, calcul de pose, etc.), nous présentons les deux grandes catégories d'algorithmes de Structure-from-Motion : le batch Structure-from-Motion, où toutes les images sont traitées simultanément, et le Structure-from-Motion incrémental ou séquentiel (SLAM), dans lequel les images sont traitées les unes après les autres.

In this chapter we present vision based algorithms for 3D reconstruction and localization, and more particularly the techniques known as Structure-from-Motion (SfM). After introducing the basic algorithms in computer vision (detection, description and matching of interest points, point triangulation, pose computation, etc..), we present the two main categories of Structure-from-Motion algorithms : the batch Structure-from-Motion, where all images are processed simultaneously, and the incremental or sequential Structure-from-Motion (SLAM), in which images are processed one after the other.

3.1 Algorithmes de base

Avant de présenter les techniques avancées de *Structure-from-Motion*, nous introduisons les différents concepts et algorithmes de base en vision par ordinateur : détection et mise en correspondance de points d'intérêts, reconstruction d'un nuage de points 3D, calcul de pose d'une caméra, estimation du déplacement inter-image et factorisation perspective.

3.1.1 Détection et mise en correspondance de points d'intérêts

Pour pouvoir localiser une caméra dans un environnement inconnu, il faut tout d'abord détecter des éléments fixes que l'on pourra facilement suivre dans les différentes images de la séquence. Ces éléments fixes, appelés éléments d'intérêts, seront dans cette thèse constitués de points 3D car ils sont omniprésents dans les types d'environnement qui nous intéressent.

Nous présentons dans cette section les différentes étapes permettant de détecter, de décrire et d'apparier les points d'intérêts 2D, les observations des points 3D dans les images. À la fin de ces trois étapes, nous obtenons un graphe de correspondances de points 2D qui sera utilisé par l'une des méthodes de *Structure-from-Motion* présentées dans la suite de ce chapitre, afin de localiser les images et de reconstruire des points 3D de la scène.

3.1.1.1 Détection de points d'intérêts

Nous nous intéressons ici à la détection de points 2D dit d'intérêts dans les images. Ces points 2D devant être facilement localisable et reconnaissable dans les différentes images, il est courant de prendre les zones saillantes ou coins (*corner*) de l'image. Les détecteurs de points d'intérêt doivent être de plus suffisamment robustes aux changements de perspective de la scène intervenant avec le déplacement de la caméra, afin de pouvoir détecter les mêmes points dans toutes les images. Il y a là une notion d'invariance de transformation 2D, par exemple l'invariance à la transformation affine [Mikolajczyk 2004, Tuytelaars 2008]. Pour cela, nous employons les points fournis par le détecteur SIFT [Lowe 2004] dans la première partie de la thèse et les points d'intérêt de HARRIS [Harris 1988] dans la seconde partie concernée par le temps réel.

Dans cette dernière, nous ne sélectionnons qu'une partie des points d'intérêts afin de gagner en temps de calcul. Pour cela nous sélectionnons un nombre fixe de points d'intérêt, les points les plus saillants grâce à l'indice de discrimination délivré par le détecteur utilisé. De plus, nous utilisons un système de baquet qui découpe l'image en sous-zones et cherchons les points d'intérêts dans ces sous-zones. Ceci permet de garantir une certaine répartition des points d'intérêt dans l'image, ce qui améliore la qualité des résultats [Zhang 1998].

3.1.1.2 Description de points d'intérêts

Afin de retrouver les observations d'un même point 3D dans une séquence d'images, il est nécessaire d'identifier le point de manière unique. Pour cela, nous employons un système de signature sur chaque points 3D. Cette signature est constituée d'un ensemble de caractéristiques, les descripteurs locaux, et permet de mesurer la similarité (ou dissimilarité) entre deux points.

Ces descripteurs doivent être suffisamment discriminants pour pouvoir différencier des points 3D différents, mais aussi suffisamment génériques pour accepter des variations entraînées soit par la scène (variations d'illumination notamment) soit par le mouvement de la caméra (changements de point de vue, variation d'échelle, etc.).

Dans cette thèse nous utilisons principalement les descripteurs SIFT [Lowe 2004] et SURF [Bay 2006]. L'inconvénient de SIFT vient de son temps de calcul, il consomme en effet beaucoup de ressource pour déterminer chaque descripteur. En comparaison, le descripteur SURF est beaucoup plus rapide (mais moins robuste que SIFT) et peut donc être employé dans des applications temps-réel. Notons qu'il existe une version temps réel du descripteur SIFT basée sur une implémentation sur GPU [Sinha 2006].

3.1.1.3 Mise en correspondance de points d'intérêts

Une fois que nous avons un ensemble de points d'intérêt avec leurs descripteurs associés, nous pouvons tenter de les mettre en correspondance : c'est l'étape d'appariement. L'objectif de cette étape est de retrouver les observations d'un même point 3D parmi l'ensemble des points d'intérêt détectés. La phase d'appariement proprement dite consiste à calculer un score de ressemblance entre chaque point d'intérêt à partir de leur descripteur, et de coupler les observations ayant les scores les plus important. L'appariement doit de plus respecter la contrainte d'unicité : un point 3D ne doit avoir au maximum qu'une seule observation par image.

Cette étape peut être affectée par de mauvais appariements, ce qui donnera au final une reconstruction peu fidèle de la réalité. Pour éviter cela, il existe des méthodes robustes (voir la section 2.3) comme par exemple le filtrage par contraintes géométriques qui utilise la matrice fondamentale pour éliminer les *outliers*, par une distance maximale à l'épipole (voir section 3.1.4).

3.1.2 Reconstruction d'un nuage de points 3D

3.1.2.1 Principe

La reconstruction d'un nuage de points 3D, aussi connu sous le nom de triangulation, consiste à déterminer les positions des points 3D à partir de leurs observations 2D (points d'intérêt) dans plusieurs images.

On s'intéressera ici seulement à la reconstruction euclidienne de points 3D à partir d'un ensemble de poses connues d'une caméra. Les techniques de reconstruction et localisation simultanées, euclidiennes ou projectives, seront détaillées dans la section 3.3.

3.1.2.2 Triangulation multivue

Le problème de la reconstruction d'une scène composée d'un ensemble de points 3D à partir de leurs observations est généralement traité en deux étapes. La première consiste à reconstruire par une méthode algébrique chaque point de la scène de manière indépendante, puis d'optimiser l'ensemble des points 3D par une technique d'optimisation.

La solution algébrique proposée par HARTLEY et STURM [Hartley 1997] formule le problème de manière à obtenir un système linéaire du type $\mathcal{A}\mathbf{Q} = \mathbf{0}$ (où $\mathbf{Q}$ est le vecteur composé des coordonnées 3D du point 3D), et résout ensuite ce système d'équations par une décomposition en valeurs singulières par exemple (voir la section 2.2.1). Mais dans la pratique le système n'est pas nul et il est alors nécessaire d'employer un algorithme de minimisation.

3.1.2.3 Triangulation stéréo

Une autre méthode de triangulation rapide est la méthode du point-milieu [Hartley 1997]. Une illustration de la méthode est proposée en Figure 3.1.

Le principe de cette méthode est de déterminer la position $\mathbf{Q}$ du point Q à partir de ses deux observations $\hat{\mathbf{q}}^1$ et $\hat{\mathbf{q}}^2$ (détectées dans deux images) en utilisant les équations de projection $\hat{\mathbf{q}}^i = \mathcal{P}_i\mathbf{Q}$. L'idée est de calculer la droite perpendiculaire commune aux droites $(\mathbf{t}_1\hat{\mathbf{q}}^1)$ et $(\mathbf{t}_2\hat{\mathbf{q}}^2)$, dont la direction $\vec{\mathbf{v}}$ est définie par $\vec{\mathbf{v}} = \overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1} \times \overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}$.

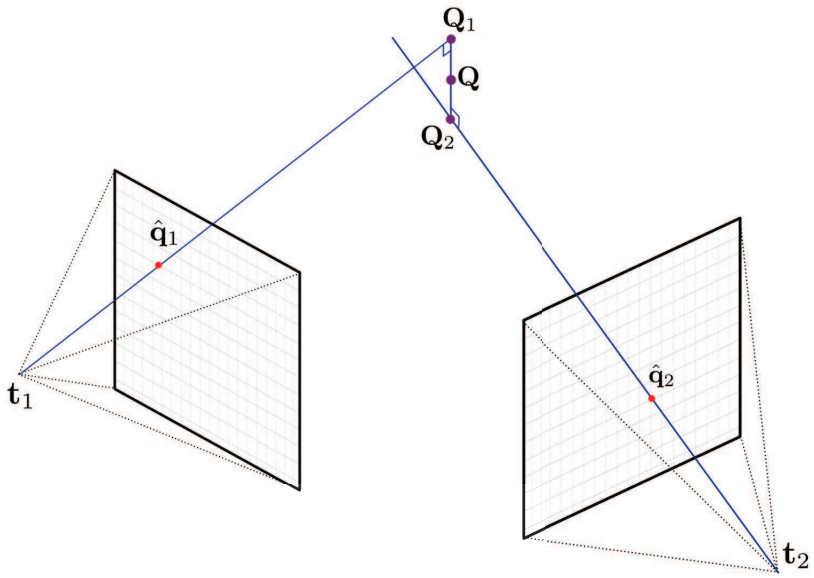


FIGURE 3.1 – Triangulation d'un point 3D à partir de deux vues.

On définit le point Q_1 de coordonnées $\mathbf{Q}_1$ comme étant l'intersection entre la droite $(\mathbf{t}_1\hat{\mathbf{q}}^1)$ et le plan $\boldsymbol{\pi}_2$. Ce plan passe par le centre optique $\mathbf{t}_2$ et possède les directions $\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}$ et $\vec{\mathbf{v}}$. De la même manière, on définit le point Q_2 de coordonnées $\mathbf{Q}_2$, intersection de la droite $(\mathbf{t}_2\hat{\mathbf{q}}^2)$ et du plan $\boldsymbol{\pi}_1$, passant par $\mathbf{t}_1$ et ayant les directions $\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}$ et $\vec{\mathbf{v}}$.

Les deux rayons optiques ainsi définis peuvent être paramétrés de la façon suivante :

$$\mathbf{Q}_1 = \mathbf{t}_1 + \alpha_1\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1} \text{ et } \mathbf{Q}_2 = \mathbf{t}_2 + \alpha_2\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}. \tag{3.1}$$

Dans la pratique, ces rayons optiques ne sont pas sécants et la méthode propose alors de trouver

le milieu du segment $\mathbf{Q}_1\mathbf{Q}_2$, défini par :

$$\mathbf{Q} = \frac{1}{2}\left((\mathbf{t}_1 + \alpha_1\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}) + (\mathbf{t}_2 + \alpha_2\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2})\right), \tag{3.2}$$

avec

$$\begin{aligned}\alpha_1 &= \frac{(\mathbf{t}_2 - \mathbf{t}_1).\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1} - (\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}.\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}) \times (\mathbf{t}_2 - \mathbf{t}_1).\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}}{1 - (\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}.\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2})^2} \\ \alpha_2 &= \frac{(\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}.\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}) \times (\mathbf{t}_2 - \mathbf{t}_1).\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1} - (\mathbf{t}_2 - \mathbf{t}_1).\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2}}{1 - (\overrightarrow{\mathbf{t}_1\hat{\mathbf{q}}^1}.\overrightarrow{\mathbf{t}_2\hat{\mathbf{q}}^2})^2}.\end{aligned} \tag{3.3}$$

On réalise le même processus pour tous les points 3D afin de reconstruire l'ensemble du nuage de points 3D que constitue la scène.

Cette technique peut aussi être employée pour reconstruire un nuage de points à partir de 3 vues (ou plus). L'idée est de calculer le point milieu pour deux couples de vues et de considérer l'isobarycentre du segment formé par les deux points milieu comme le point 3D recherché.

3.1.2.4 Triangulation optimale

Une fois que les positions de tous les points 3D de la scène sont estimées par une des deux méthodes algébriques mentionnées, une étape d'optimisation est généralement employée afin de limiter les erreurs introduites par la méthode précédente. Pour cela nous faisons appel à un algorithme de minimisation non-linéaire (section 2.2.2) dans lequel nous optimisons simultanément tous les points 3D, et minimisons l'erreur de reprojection (voir section 3.2.2) : la distance entre les coordonnées de l'observation $\hat{\mathbf{q}}_j$ et la reprojection théorique (en $\mathbf{q}_j = \mathcal{P}\mathbf{Q}_j$) du point 3D dans l'image. Nous résolvons le problème suivant :

$$[\mathbf{Q}_1^*, \ldots, \mathbf{Q}_{N_n}^*] = \arg \min_{[\mathbf{Q}_1,\ldots,\mathbf{Q}_{N_n}]} F^c([\mathbf{P}\mathbf{Q}_1 \ldots \mathbf{Q}_{N_n}]), \tag{3.4}$$

où $F^c(\mathbf{x})$ est la fonction de coût de reprojection géométrique définie dans la section 3.2.2.

La résolution de l'équation (3.4) est effectuée par un des algorithmes de minimisation non-linéaire présentés dans la section 2.2.2, par exemple avec l'algorithme de LEVENBERG-MARQUARDT.

3.1.3 Calcul de pose d'une caméra

Cette section traite de l'obtention des paramètres extrinsèques d'une caméra à partir d'un nuage de points 3D (déjà reconstruit) et de leurs observations détectées dans l'image. Ce problème est aussi connu sous le nom de resection ou de problème PnP (*Perspective-n-Points*). Plus précisément, nous cherchons à estimer les 6 degrés de liberté de la pose (3 degrés pour la position $\mathbf{t}$ et 3 degrés pour l'orientation $\mathcal{R}$) à partir des correspondances 3D/2D : $\mathbf{Q}_j \leftrightarrow \hat{\mathbf{q}}_j$.

Pour résoudre ce problème, plusieurs solutions ont été proposées [Grünert 1841, Haralick 1994]. Ces méthodes varient selon le nombre de points utilisés et sont plus ou moins

robustes aux mauvais appariements et aux erreurs de positionnement des points d'intérêts. Le principe général de ces méthodes est de trouver une estimation initiale robuste de la pose et de raffiner ensuite cette solution à l'aide d'une optimisation non-linéaire.

Nous présentons la méthode que nous avons utilisée dans cette thèse, soit le calcul de pose à partir de 3 points par la méthode de GRÜNERT [Grünert 1841].

3.1.3.1 Calcul de pose par la méthode des 3 points

Chaque paire de correspondances 3D/2D (un premier point 3D $\mathbf{Q}_j$ et son observation 2D $\hat{\mathbf{q}}_j$, un second point 3D $\mathbf{Q}_k$ et son observation 2D $\hat{\mathbf{q}}_k$) fournit une contrainte sur les distances (aussi appelées profondeurs) entre les points 3D et le centre optique $\mathbf{t}$ de la caméra. Ces distances, notées $x_j = \|\mathbf{Q}_j - \mathbf{t}\|$ et $x_k = \|\mathbf{Q}_k - \mathbf{t}\|$ doivent être premièrement estimées afin d'obtenir la pose finale de la caméra. La contrainte de la paire s'écrit (théorème d'Al-Kashi) :

$$d_{jk}^2 = x_j^2 + x_k^2 - 2x_j x_k \cos\theta_{jk}, \tag{3.5}$$

où d_{jk} représente la distance entre les deux points 3D $\mathbf{Q}_j$ et $\mathbf{Q}_k$ et θ_{jk} est l'angle entre les deux rayons optiques passant par le centre optique de la caméra et par les observations, respectivement $\hat{\mathbf{q}}_j$ et $\hat{\mathbf{q}}_k$. Cet angle peut être calculé par un simple produit scalaire des vecteurs directeurs normés $\mathbf{d}_j$ et $\mathbf{d}_k$: $\cos\theta_{jk} = \mathbf{d}_j^\top \mathbf{d}_k$.

Une illustration de la configuration est donnée dans la figure 3.2.

L'utilisation de trois points 3D génère ainsi trois contraintes qui sont ensuite réécrites ($f_{jk} = x_j^2 + x_k^2 - 2x_j x_k \cos\theta_{jk} - d_{jk}^2 = 0$) pour former un système d'équations quadratiques :

$$\begin{cases} f_{jk}(x_j, x_k) = 0 \\ f_{jl}(x_j, x_l) = 0 \\ f_{kl}(x_k, x_l) = 0 \end{cases}. \tag{3.6}$$

Nous cherchons alors les 3 profondeurs (x_j, x_k, x_l) des 3 points 3D par la méthode proposée par GRÜNERT [Grünert 1841]. L'idée de GRÜNERT est d'utiliser un changement de variable pour réduire le nombre d'inconnues du système et ainsi obtenir un polynôme de degré 4. Les 4 solutions du polynôme sont alors calculées de manière explicite pour obtenir au final 4 triplets possibles de distances (x_j, x_k, x_l).

Enfin, pour extraire les informations de pose de la caméra à partir de ces distances, nous avons utilisé la technique de FINSTERWALDER *et al.* [Finsterwalder 1937]. D'autres solutions existent et ont été comparées par HARALICK *et al.* [Haralick 1994] et plus récemment par LEPETIT *et al.* [Lepetit 2009]. L'estimation robuste nous permettra par la suite de sélectionner la bonne solution parmi les multiples solutions (au maximum 4) ainsi générées.

3.1.3.2 Calcul de pose robuste

Les principales difficultés rencontrées lors du calcul de pose sont de deux types. D'une part certaines associations 3D/2D peuvent être erronées, par exemple à cause d'une erreur lors de la

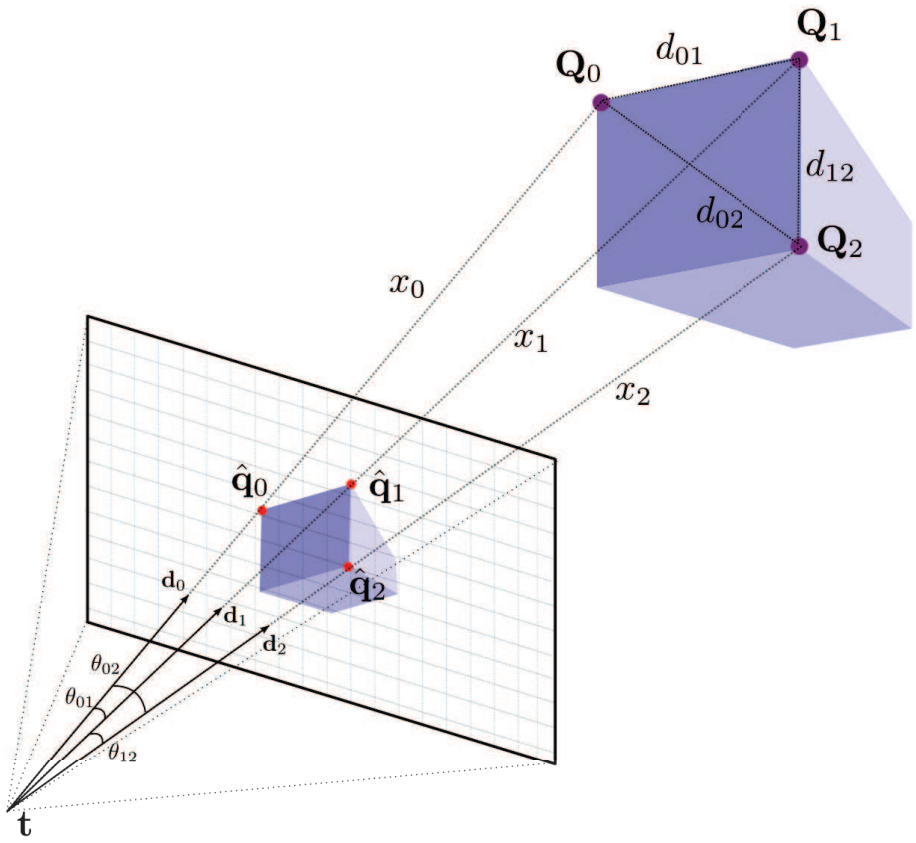


FIGURE 3.2 – Calcul de pose à partir des correspondances 3D/2D.

mise en correspondance : lorsque les descripteurs sont proches et peuvent donc être confondus. Et d'autre part lorsque la détection des points d'intérêts fournit des positions 2D approximatives des observations. Pour obtenir une pose précise, il est très utile d'avoir une détection sous-pixellique des points d'intérêt, qui est bien plus précise qu'une simple détection délivrant des coordonnées en pixels.

Pour pallier ces difficultés, nous employons la technique RANSAC (voir section 2.3). Le principe est d'identifier un triplet de points robuste (que l'on sait être fiables) d'une part, et d'estimer la pose de la caméra sur l'ensemble des points dont nous avons retiré les données aberrantes. Plus précisément, le calcul de pose robuste par RANSAC se compose des étapes suivantes :

- On effectue un tirage aléatoire de 3 points 3D (Q_j, Q_k, Q_l) parmi l'ensemble des points du nuage.
- On estime ensuite la pose de la caméra à l'aide de la méthode précédente et nous obtenons une ou plusieurs solutions (1 ou 2, voire plus rarement 4).
- Pour chaque solution, on compte le nombre d'appariements cohérents. Un appariement est dit cohérent (*inlier*) lorsque l'erreur de reprojection du point 3D (définie en section 3.2.2) est inférieure à un seuil (par exemple 3 pixels).

– On choisit enfin l'échantillon ayant le plus grand nombre d'appariements cohérents et la pose associée.

Cette méthode permet d'estimer la pose de manière robuste et élimine ainsi les valeurs aberrantes (*outliers*).

3.1.3.3 Calcul de la pose optimale

À partir de la solution initiale robuste ainsi estimée, nous pouvons entreprendre une minimisation non linéaire dans le but d'affiner la pose de la caméra. Nous cherchons donc à optimiser les paramètres de la pose (la position $\tilde{\mathbf{t}}$ et l'orientation, paramétrée par exemple par la matrice de rotation $\mathcal{R}$) :

$$[\mathbf{R}^*, \tilde{\mathbf{t}}^*] = \arg\min_{[\mathbf{R},\tilde{\mathbf{t}}]} F^c([\mathbf{R}\tilde{\mathbf{t}}\mathbf{Q}_1 \dots \mathbf{Q}_{N_n}]), \tag{3.7}$$

où $\mathbf{R} = \text{vect}(\mathcal{R})$ et F^c est la fonction de reprojection géométrique définie dans la section 3.2.2. Nous verrons que ces fonctions de reprojection sont souvent non linéaires et la résolution de l'équation (3.7) doit être effectuée par un des algorithmes de minimisation non linéaire présentés dans la section 2.2.2, par exemple avec l'algorithme de LEVENBERG-MARQUARDT.

Cette optimisation estime donc la pose optimale de la caméra au sens des moindres carrés, mais elle est aussi une optimisation robuste puisqu'elle bénéficie de la détection des *inliers* grâce à RANSAC. Il est à noter que d'autres optimisations robustes existent, comme par exemple les *M-estimateur* (voir section 2.3.2).

3.1.4 Estimation du déplacement inter-image

Cette section traite de l'estimation du déplacement relatif entre plusieurs images. Ce problème peut être résolu par différentes techniques pour deux images [Longuet-Higgins 1981, Hartley 1995, Torr 1997, Nistér 2004a, Sarkis 2007], 3 images [Hartley 2003b] ou pour un nombre supérieur d'images N_m (voir la section 3.3). Nous commençons par rappeler les concepts de la géométrie épipolaire puis nous présentons l'estimation du déplacement entre deux images par la méthode des 5 points.

3.1.4.1 Géométrie épipolaire

La géométrie épipolaire regroupe l'ensemble des relations géométriques s'exerçant entre deux images, et ce de manière indépendante de la scène rigide filmée. Le concept de géométrie épipolaire est aujourd'hui bien connu de la communauté vision (voir notamment les livres [Faugeras 1993] et [Hartley 2003b]) et nous en rappelons ici quelques concepts : les matrices fondamentales et essentielles.

Matrice fondamentale. Supposons que nous disposions de deux observations $\hat{\mathbf{q}}_1$ et $\hat{\mathbf{q}}_2$ (exprimés en coordonnées homogènes) d'un même point 3D Q de coordonnées homogènes $\mathbf{Q}$. Ces deux observations correspondent par exemple à deux points d'intérêt détectés respectivement

dans les caméras c_1 et c_2 par le procédé présenté dans la partie 3.1.1. Grâce à la géométrie épipolaire, nous savons qu'un point $\hat{\mathbf{q}}_1$ se situe sur une ligne épipolaire $\mathbf{l}_2 \sim \mathcal{F}\hat{\mathbf{q}}_1$ dans la deuxième caméra (voir la figure 3.3). De plus, si $\hat{\mathbf{q}}_1$ et $\hat{\mathbf{q}}_2$ sont les observations d'un même point 3D, alors elles vérifient la relation suivante :

$$\hat{\mathbf{q}}_2^\top \mathcal{F} \hat{\mathbf{q}}_1 = 0. \tag{3.8}$$

La matrice fondamentale $\mathcal{F}$ est une matrice 3×3 de rang 2. Elle ne dépend que des paramètres intrinsèques et extrinsèques des deux caméras et est par conséquent indépendante de la scène. Les épipoles $\mathbf{e}_1$ et $\mathbf{e}_2$ correspondent à la projection des centres optiques dans les deux images. Ces points particuliers forment le noyau de la matrice fondamentale $\mathcal{F}\mathbf{e}_1 = 0$ et $\mathcal{F}^\top \mathbf{e}_2 = 0$.

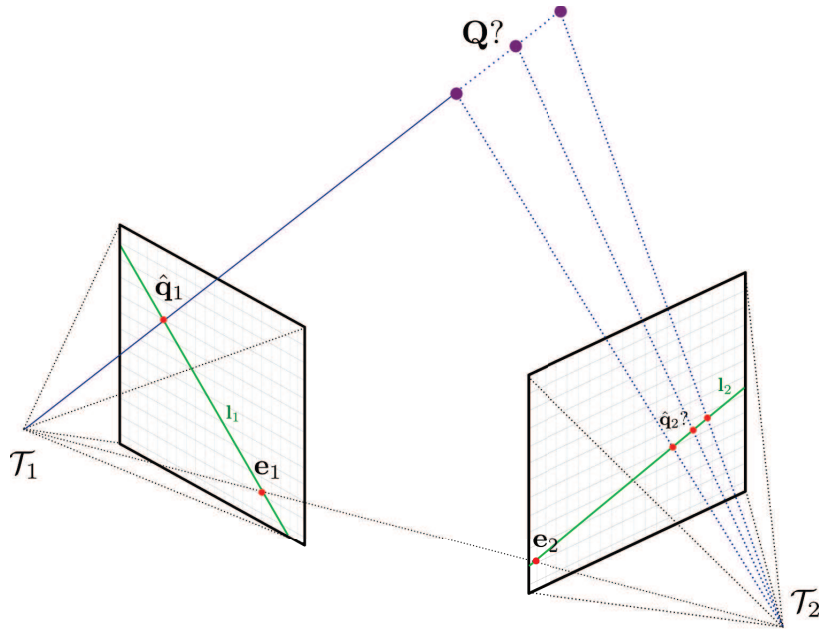


FIGURE 3.3 – Géométrie épipolaire.

La matrice fondamentale est une entité très utile car elle peut être utilisée pour retrouver les matrices de projections $\mathcal{P}_i$ des caméras. Puisque nous cherchons ici à estimer le déplacement métrique entre deux caméras, il n'est pas nécessaire d'estimer cette matrice, nous calculons la matrice essentielle. Lorsque les caméras sont calibrées, c'est-à-dire que l'on connaît précisément leurs paramètres intrinsèques, nous pouvons estimer la matrice essentielle entre ces deux caméras à partir de la matrice fondamentale.

Matrice essentielle. À l'image de la matrice fondamentale, la matrice essentielle introduite par LONGUET-HIGGINS [Longuet-Higgins 1981], relie deux points homologues $\hat{\mathbf{d}}_1$ et $\hat{\mathbf{d}}_2$, exprimés cette fois dans le plan principal de chaque caméra (et non dans le repère image) : $\hat{\mathbf{d}}_i = \mathcal{K}_i^{-1}\hat{\mathbf{q}}_i$. $\mathcal{K}_i$ est la matrice de calibrage de la caméra c_i.

La matrice essentielle définit la relation bilinéaire suivante :

$$\hat{\mathbf{d}}_2^\top \mathcal{E} \hat{\mathbf{d}}_1 = 0, \tag{3.9}$$

où $\mathcal{E} \sim \mathcal{K}_2^\top \mathcal{F} \mathcal{K}_1$ est la matrice essentielle. Celle-ci ne possède que cinq degrés de liberté qui correspondent aux paramètres extrinsèques du déplacement inter-image, soit 6 degrés de liberté moins un à cause de l'ambiguïté du facteur d'échelle.

3.1.4.2 Calcul de déplacement inter-image par la méthode des 5 points

Dans cette section nous nous intéressons à l'estimation du déplacement entre deux images dans le cas où les caméras sont calibrées. Dans ces conditions, il n'est pas nécessaire de calculer la matrice fondamentale par les algorithmes de l'état de l'art [Longuet-Higgins 1981, Hartley 1995, Torr 1997]. En effet, nous pouvons directement estimer la matrice essentielle par la méthode dite des 5 points [Nistér 2004a]. Cette méthode a l'avantage d'être plus efficace puisque la matrice essentielle possède moins de degrés de liberté à calculer (5, contre 7 pour la matrice fondamentale).

Méthode des 5 points. La méthode des 5 points [Nistér 2004a, Sarkis 2007] est une méthode efficace et minimale pour estimer le déplacement inter-image à partir de correspondances de points d'intérêt. L'équation (3.9) peut être réécrite sous la forme d'une équation linéaire, telle que :

$$\mathbf{a}^\top \mathbf{e} = 0, \tag{3.10}$$

où $\mathbf{e}$ est un vecteur de taille 9×1 contenant les coefficients de $\mathcal{E}$ tel que $\mathbf{e} = [\mathcal{E}_{00}\mathcal{E}_{01}\mathcal{E}_{02}\mathcal{E}_{10}\mathcal{E}_{11}\mathcal{E}_{12}\mathcal{E}_{20}\mathcal{E}_{21}\mathcal{E}_{22}]$ et $\mathbf{a}$ contient les combinaisons des coordonnées de $\hat{\mathbf{d}}_1 = (\hat{d}_1^x, \hat{d}_1^y, \hat{d}_1^w)$ et $\hat{\mathbf{d}}_2 = (\hat{d}_2^x, \hat{d}_2^y, \hat{d}_2^w)$:

$$\mathbf{a} = [\hat{d}_1^x\hat{d}_2^x, \hat{d}_1^x\hat{d}_2^y, \hat{d}_1^x\hat{d}_2^w, \hat{d}_1^y\hat{d}_2^x, \hat{d}_1^y\hat{d}_2^y, \hat{d}_1^y\hat{d}_2^w, \hat{d}_1^w\hat{d}_2^x, \hat{d}_1^w\hat{d}_2^y, \hat{d}_1^w\hat{d}_2^w]^\top. \tag{3.11}$$

Chacune des 5 correspondances fournit donc un vecteur $\mathbf{a}_i$ différent et vérifie l'équation (3.10). Nous pouvons alors empiler ces contraintes linéaires sous la forme d'un système :

$$\mathcal{A}_{5\times 9}^\top \mathbf{e} = 0. \tag{3.12}$$

La solution de ce système est l'espace de dimensions 4 formé par la base $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4)$ du noyau de $\mathcal{A}_{5\times 9}$. La solution $\mathbf{e}$ doit par conséquent être de la forme

$$\mathbf{e} = x\mathbf{e}_1 + y\mathbf{e}_2 + z\mathbf{e}_3 + w\mathbf{e}_4, \tag{3.13}$$

où $x, y, z, w \in \mathbb{R}$. Comme la matrice essentielle est définie à un coefficient multiplicatif près, on fixe le coefficient w à 1. Il reste alors encore trois inconnues à calculer pour obtenir la matrice essentielle. Aux 5 équations précédentes, nous ajoutons une contrainte modélisant le fait que la matrice essentielle possède deux valeurs singulières identiques et une troisième nulle. $\mathcal{E}$ doit par conséquent vérifier l'équation :

$$\mathcal{E}\mathcal{E}^\top\mathcal{E} - \frac{1}{2}\mathrm{tr}(\mathcal{E}\mathcal{E}^\top)\mathcal{E} = \mathbf{0}. \tag{3.14}$$

Les deux équations (3.12) et (3.14) nous donne alors 9 contraintes linéaires et non linéaires et 3 inconnues à déterminer. Ensuite, par l'utilisation notamment de l'élimination de GAUSS-JORDAN nous sommes en mesure de calculer les solutions du triplet x, y, z et au final de calculer la matrice essentielle. Le détail de ces calculs n'est pas développé ici et nous renvoyons le lecteur intéressé à l'article de NISTER [Nistér 2004a] ainsi qu'à la thèse de ROYER [Royer 2006].

Extraction des paramètres extrinsèques à partir de la matrice essentielle. Comme nous venons de le voir, la matrice essentielle n'est composée que des paramètres de déplacement entre deux caméras (paramètres extrinsèques). En considérant les matrices de poses des deux caméras suivantes :

$$\mathcal{C}_1 = \begin{pmatrix} \mathcal{I}_3 & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{pmatrix} \quad \text{et} \quad \mathcal{C}_2 = \begin{pmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \tag{3.15}$$

la matrice essentielle peut se décomposer ainsi :

$$\mathcal{E} = [\mathbf{t}]_\times \mathcal{R} = \mathcal{R}[\mathcal{R}^\top \mathbf{t}]_\times. \tag{3.16}$$

avec $[\mathbf{t}]_\times$ la matrice antisymétrique de $\mathbf{t}$. L'extraction des paramètres extrinsèques de $\mathcal{E}$ s'opère de la manière suivante. On décompose la matrice essentielle en valeurs singulières (SVD) et on obtient alors $\mathcal{E} = \mathcal{U} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \mathcal{V}$, avec $\det(\mathcal{U}) > 0$ et $\det(\mathcal{V}) > 0$. En effet, une des propriétés de la matrice essentielle est d'avoir ses deux premières valeurs singulières égales et la troisième nulle (elle est de rang 2). À partir de cette décomposition de $\mathcal{E}$, nous obtenons 4 solutions possibles de $\mathcal{C}_2$ qui sont [Hartley 2003b] :

$$\mathcal{C}_2 = \begin{pmatrix} \mathcal{U}\mathcal{W}\mathcal{V}^\top & +\mathbf{u}_3 \\ \mathbf{0}^\top & 1 \end{pmatrix}, \begin{pmatrix} \mathcal{U}\mathcal{W}\mathcal{V}^\top & -\mathbf{u}_3 \\ \mathbf{0}^\top & 1 \end{pmatrix}, \begin{pmatrix} \mathcal{U}\mathcal{W}^\top\mathcal{V}^\top & +\mathbf{u}_3 \\ \mathbf{0}^\top & 1 \end{pmatrix}, \begin{pmatrix} \mathcal{U}\mathcal{W}^\top\mathcal{V}^\top & -\mathbf{u}_3 \\ \mathbf{0}^\top & 1 \end{pmatrix}. \tag{3.17}$$

avec $\mathcal{W} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ et $\mathbf{u}_3$ est la troisième colonne de la matrice $\mathcal{U}$.

Pour choisir la bonne solution de $\mathcal{C}_2$, nous faisons tout d'abord l'hypothèse que celle-ci est parmi les deux premières possibilités (nous choisissons $\mathcal{R}_2 = \mathcal{U}\mathcal{W}\mathcal{V}^\top$). Ensuite, le principe est de reconstruire un point avec les deux possibilités restantes pour lever l'ambiguïté. En effet, la solution recherchée est alors celle dont le point reconstruit est devant les deux caméras.

3.1.4.3 Calcul de déplacement robuste et optimal

À la manière du calcul de pose robuste (section 3.1.3.2), il est souvent utile de réduire l'impact des faux appariements sous peine d'avoir une estimation erronée du déplacement. Pour cela nous employons la technique RANSAC (section 2.3.1) qui se compose des étapes suivantes :

- On effectue un tirage aléatoire de 5 correspondances
- On estime ensuite le déplacement inter-image à l'aide de la méthode des 5 points

– On reconstruit tous les points 3D à partir de l'estimation du déplacement ainsi obtenu et des correspondances 2D
– On compte le nombre d'appariements cohérents (c'est-à-dire ceux dont l'erreur de reprojection du point 3D est inférieure à un seuil de 4 pixels par exemple).
– Enfin, on choisit le déplacement ayant le plus grand nombre d'appariements cohérents (*inliers*) et le déplacement associé.

Cette méthode permet d'estimer le déplacement de manière robuste et élimine ainsi les faux appariements (*outliers*). Enfin, nous optimisons le déplacement inter-image (plus précisément, la pose $\mathcal{C}_2$) à l'aide d'un algorithme de minimisation non linéaire (voir section 2.2.2) en résolvant l'équation (3.7). Ce dernier raffinement nous permet d'avoir une estimation robuste et optimale.

3.1.5 Factorisation perspective

3.1.5.1 Formulation du problème

La reconstruction 3D à partir d'images non calibrées peut être effectuée par différents moyens : la matrice fondamentale (2 images), les tenseurs trifocaux (3 images) et quadrifocaux (4 images) [Hartley 2003b]. Avec un nombre quelconque d'images, la techniques de factorisation peut être employée. La factorisation a été initialement proposée pour des caméras orthographiques [Kanade 1992] puis elle a été étendue pour des caméras perspectives [Sturm 1996, Hartley 2003a].

La projection d'un point 3D de coordonnées homogènes $\mathbf{Q}_j$ effectuée par une caméra perspective c_i (de matrice de projection $\mathcal{P}_i$) peut être mise sous la forme suivante :

$$\lambda_j^i \hat{\mathbf{q}}_j^i = \mathcal{P}_i \mathbf{Q}_j, \tag{3.18}$$

où λ_j^i est la *profondeur projective* et les coordonnées de l'observation $\hat{\mathbf{q}}_j^i$ sont homogènes avec une troisième coordonnée égale à 1. Les projections des N_n points 3D dans les N_m images peuvent être combinées sous la forme du système suivant :

$$\underbrace{\begin{pmatrix} \lambda_1^1\hat{\mathbf{q}}_1^1 & \lambda_2^1\hat{\mathbf{q}}_2^1 & \dots & \otimes \\ \lambda_1^2\hat{\mathbf{q}}_1^2 & \otimes & \dots & \lambda_{N_n}^2\hat{\mathbf{q}}_{N_n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \otimes & \lambda_2^{N_m}\hat{\mathbf{q}}_2^{N_m} & \dots & \lambda_{N_n}^{N_m}\hat{\mathbf{q}}_{N_n}^{N_m} \end{pmatrix}}_{\mathcal{M}_{3N_m\times N_n}} = \underbrace{\begin{pmatrix} \mathcal{P}_1 \\ \mathcal{P}_2 \\ \vdots \\ \mathcal{P}_{N_m} \end{pmatrix}}_{\mathcal{P}_{3N_m\times 4}} \underbrace{\begin{pmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 & \dots & \mathbf{Q}_{N_n} \end{pmatrix}}_{\mathcal{Q}_{4\times N_n}}, \tag{3.19}$$

où $\otimes$ indique qu'une observation n'a pu être effectuée à cause par exemple d'une occultation, $\mathcal{P}_{3N_m\times 4}$ et $\mathcal{Q}_{4\times N_n}$ représentent respectivement les mouvements de la caméra et la structure de la scène (points 3D). La matrice $\mathcal{M}$ est appelée matrice de mesure mise à l'échelle ou *rescaled measurement matrix*.

L'objectif de la factorisation est alors de retrouver les matrices de projection ($\mathcal{P}_{i\in[1..N_m]}$) et la structure de la scène ($\mathcal{Q}_{j\in[1..N_n]}$) à partir de cette matrice de mesure.

Pour cela, les techniques [Triggs 1996, Sturm 1996, Heyden 1997, Martinec 2005] intégrant la factorisation se composent notamment des étapes suivantes :

1. Normaliser les observations 2D par une normalisation isotropique $\mathcal{N}_i$ (section 4.2.2)
2. Estimer les matrices fondamentales et leurs épipoles (section 3.1.4)
3. Estimer des profondeurs projectives (section 3.1.5.2)
4. Mettre à jour la matrice de mesure mise à l'échelle avec les profondeurs estimées
5. Calculer une décomposition SVD de la matrice de mesure (ou des sous-matrices sans données manquantes, voir section 3.1.5.4) et en déduire les matrices de projection des caméras et la structure de la scène
6. Rectifier les matrices de projection pour intégrer la dénormalisation $\mathcal{P}'_i = \mathcal{N}_i^{-1}\mathcal{P}_i$
7. (Optionnel) Reprojeter les points 3D dans chaque image pour obtenir une nouvelle profondeur projective et répéter les étapes à partir de (4)

3.1.5.2 Estimation des profondeurs projectives

Une difficulté importante de la factorisation est l'estimation des profondeurs projectives. Parmi les solutions proposées, nous pouvons mentionner les techniques basées sur la géométrie épipolaire [Sturm 1996, Martinec 2005]. Une autre alternative consiste à choisir la valeur initiale des profondeurs, par exemple l'unité, et d'améliorer itérativement ces estimations à l'aide de l'erreur de reprojection, par exemple [Heyden 1997].

Dans cette thèse nous avons utilisé la technique d'estimation des profondeurs projectives de MARTINEC *et al.* [Martinec 2005], qui est une extension de la méthode de STURM et TRIGGS [Sturm 1996]. L'estimation des profondeurs s'effectue à l'aide des matrices fondamentales inter-caméra (entre les caméras i et k) et des contraintes sur les lignes épipolaires :

$$(\mathcal{F}_{ik}\mathbf{q}_j^k)\lambda_j^k = (\mathbf{e}_{ik} \times \mathbf{q}_j^i)\lambda_j^i. \tag{3.20}$$

La stratégie d'estimation des profondeurs projectives est alors directe. L'équation (3.20) relie les profondeurs d'un point 3D dans deux images et, si l'on possède un nombre suffisant de matrices fondamentales et leurs épipoles (en utilisant la méthodes des 7 points [Torr 1997] par exemple), alors nous pouvons former un système linéaire et ainsi obtenir les profondeurs projectives de l'ensemble des points 3D (à un facteur multiplicatif commun près). La résolution de l'équation (3.20) par la méthodes de moindres carrés de λ_j^i nous donne la relation suivante :

$$\lambda_j^i = \frac{(\mathbf{e}_{ik} \times \mathbf{q}_j^i)^\top (\mathcal{F}_{ik}\mathbf{q}_j^k)}{\|\mathbf{e}_{ik} \times \mathbf{q}_j^i\|^2}\lambda_j^k. \tag{3.21}$$

En initialisant arbitrairement les profondeurs projectives pour la première caméra à l'unité ($\lambda_j^1 = 1, \forall j \in [1..N_n]$), l'équation (3.21) définit de manière incrémentale les profondeurs des projections suivantes. Nous obtenons ainsi l'ensemble des profondeurs projectives, définies à un facteur près.

3.1.5.3 Résolution par décomposition en valeurs singulières

Sous l'hypothèse que les profondeurs projectives λ_j^i sont connues et que toutes les projections ont été observées, les matrices de projection et la structure de la scène peuvent être estimées par une factorisation de rang 4 de la matrice de mesures (ou de rang 3 s'il s'agit d'une caméra affine).

Une solution simple consiste à décomposer la matrice de mesure en valeurs singulières (SVD) et de forcer celle-ci à être de rang 4 (en ne gardant que ses 4 plus grandes valeurs propres). Nous obtenons $\mathcal{M} = \mathcal{U}\mathcal{S}'\mathcal{V}$, où $\mathcal{S}'$ est la matrice diagonale de la décomposition SVD de $\mathcal{M}$ avec seulement les 4 plus grandes valeurs propres. On obtient alors une solution du problème (3.19) avec $\mathcal{P} = \mathcal{U}\mathcal{S}'$ et $\mathcal{Q} = \mathcal{V}$.

Cette solution n'est pas unique. Il existe en effet un ensemble de solutions définies à une transformation projective $\mathcal{H}_{4\times4}$ près, puisque :

$$\mathcal{M}_{3N_m\times N_n} = \mathcal{P}_{3N_m\times4}\mathcal{H}\mathcal{H}^{-1}\mathcal{Q}_{4\times N_n} = \mathcal{P}_{3N_m\times4}\mathcal{Q}_{4\times N_n}. \tag{3.22}$$

Nous parlons alors d'une reconstruction projective de la scène. La partie 3.3.1 décrit les étapes nécessaires pour retrouver une reconstruction métrique de la scène (à un facteur d'échelle près) à partir de cette première reconstruction projective.

3.1.5.4 Gestion des occultations

Une limitation de la factorisation, très contraignante en pratique, est que tous les points 3D que l'on souhaite reconstruire doivent être visibles dans toutes les images. Dans la réalité, les occultations ne nous permettent pas de garantir cette hypothèse et nous nous retrouvons alors avec de nombreuses zones vides dans la matrice de mesure.

Ce problème a été évoqué par TOMASI et KANADE [Kanade 1992] puis résolu de manière automatique pour des caméras orthographiques [Jacobs 1997] et perspectives [Sturm 1996, Hartley 2003a, Guilbert 2006, Tardif 2007].

L'idée générale de ces méthodes [Sturm 1996, Hartley 2003a, Guilbert 2006, Tardif 2007] est de rechercher dans la matrice de mesure des sous-matrices pleines (sans données manquantes) et dont les profondeurs projectives ont été estimées. Ensuite, des contraintes sur les matrices de projection restant à estimer (*Camera Basis Constraints*) sont générées avec les décompositions SVD de ces sous-matrices. Un système linéaire sur les paramètres de projection est alors formé. La résolution de ce système peut être ensuite effectuée soit en estimant des vecteurs orthogonaux de dimension 4 [Martinec 2005], soit en fixant une des bases à l'identité puis en résolvant le système [Tardif 2007]. Les auteurs de [Tardif 2007] ont étendu ces contraintes sur la structure (*Structure Basis Constraints*) et ont proposé une solution au problème de fermeture de boucles dans la factorisation (*Camera and Structure Closure Constraints*).

Une fois que les matrices de projection sont ainsi estimées, un algorithme de triangulation des points 3D est alors appelé afin de reconstruire la scène.

Certains auteurs [Jacobs 1997] proposent alors de remplir les mesures manquantes par les projections des points 3D, puis en factorisant la matrice de mesure, en itérant éventuellement le processus. Cependant, lors de ce remplissage, de nouvelles erreurs sont introduites dues aux estimations approximatives des matrices de projection et de la scène reconstruite. La décomposition

SVD est alors biaisée par ces erreurs et ne fournit pas forcément une meilleure solution que sans le remplissage.

3.2 Ajustement de faisceaux

L'ajustement de faisceaux est un processus d'optimisation qui, à partir d'une première estimation du modèle à optimiser (comprenant les paramètres de la caméra et la structure de la scène), raffine celle-ci à l'aide d'un algorithme d'optimisation non linéaire, comme par exemple l'algorithme de LEVENBERG-MARQUARDT. On distinguera l'ajustement de faisceaux calibré (*euclidean bundle adjustment*) de l'ajustement de faisceaux non-calibré (*projective bundle adjustment*) par le fait que les paramètres intrinsèques de la caméra sont connus dans le premier cas et qu'il n'est donc pas nécessaire de les optimiser. Une synthèse complète sur l'ajustement de faisceaux est disponible dans [Triggs 2000].

3.2.1 Énoncé du problème

La puissance de l'ajustement de faisceaux est d'optimiser simultanément la trajectoire de la caméra et la structure de la scène. Ainsi, les N_p paramètres du modèle (le vecteur $\mathbf{x} \in \mathbb{R}^{N_p}$) sont composés des paramètres de projection $\mathcal{P}_{i=1\ldots N_m}$ de la caméra pour les N_m images et des coordonnées homogènes des N_n points 3D $\mathbf{Q}_{j=1\ldots N_n}$ de la scène dans le cas non calibré, et en coordonnées affines dans les ajustement de faisceaux calibrés.

Les paramètres de la caméra à raffiner dépendent de l'application dans laquelle est intégré l'ajustement de faisceaux et du type d'ajustement de faisceaux considéré. Dans l'ajustement de faisceaux non-calibré, nous optimisons l'ensemble des 12 paramètres de la matrice de projection $\mathcal{P}_i$ de chaque pose. Notons ici la redondance du 12ème paramètre puisque la matrice de projection ne comprend que 11 degrés de liberté. En revanche, avec l'ajustement de faisceaux calibré nous pouvons réduire ce nombre en n'optimisant qu'une partie des paramètres (extrinsèques et intrinsèques) de la pose. Dans cette thèse nous considérons les deux types d'ajustement de faisceaux : calibrés et non calibrés. Le vecteur $\mathbf{x} \in \mathfrak{M}$ des paramètres du modèle est le suivant :

$$\mathbf{x} = \begin{bmatrix} \mathbf{P}_1 \\ \vdots \\ \mathbf{P}_{N_n} \\ \mathbf{Q}_1 \\ \vdots \\ \mathbf{Q}_{N_m} \end{bmatrix} \quad \text{avec} \quad \begin{cases} \mathbf{P}_i = \text{vect}(\mathcal{P}_i), \text{ dans le cas non-calibré} \\ \mathbf{P}_i = \begin{bmatrix} \mathbf{R}^i = \text{vect}(\mathcal{R}^i) \\ \tilde{\mathbf{t}}^i \end{bmatrix} \text{ , dans le cas calibré} \end{cases} . \tag{3.23}$$

L'espace des paramètres du modèle $\mathbf{x}$ est un espace $\mathfrak{M}$ de grandes dimensions puisque ce vecteur peut compter des milliers de points 3D et des centaines de poses dans des ajustements de faisceaux globaux.

L'ajustement de faisceaux est un problème d'estimation de paramètres comme décrit dans la section 2.1. L'objectif est de trouver les paramètres du modèle $\mathbf{x} \in \mathfrak{M}$, c'est-à-dire la tra-

jectoire de la caméra et la structure de la scène qui maximisent la probabilité d'avoir observé l'ensemble des points d'intérêt $\hat{\mathbf{q}}_j^i$ détectés dans les images. On peut montrer que l'ajustement de faisceaux revient à minimiser une fonction de coût constituée des erreurs de reprojection, et nous retrouvons alors l'équation (2.12), que nous réécrivons en :

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathfrak{M}} F^c(\mathbf{x}), \tag{3.24}$$

où $F^c(\mathbf{x})$ est la fonction de coût comprenant les erreurs de reprojection définies ci-après.

3.2.2 Erreurs de reprojection

L'erreur de reprojection est une mesure statistique caractérisant l'adéquation entre le modèle et les observations du modèle. Exprimée en pixels, l'erreur de reprojection est la somme des distances entre les observations 2D ($\hat{\mathbf{q}}_j^i$) et les reprojections théoriques $\mathbf{q}_j^i$ des points 3D sur les images. Une représentation de l'erreur de reprojection est donnée en Figure 3.4.

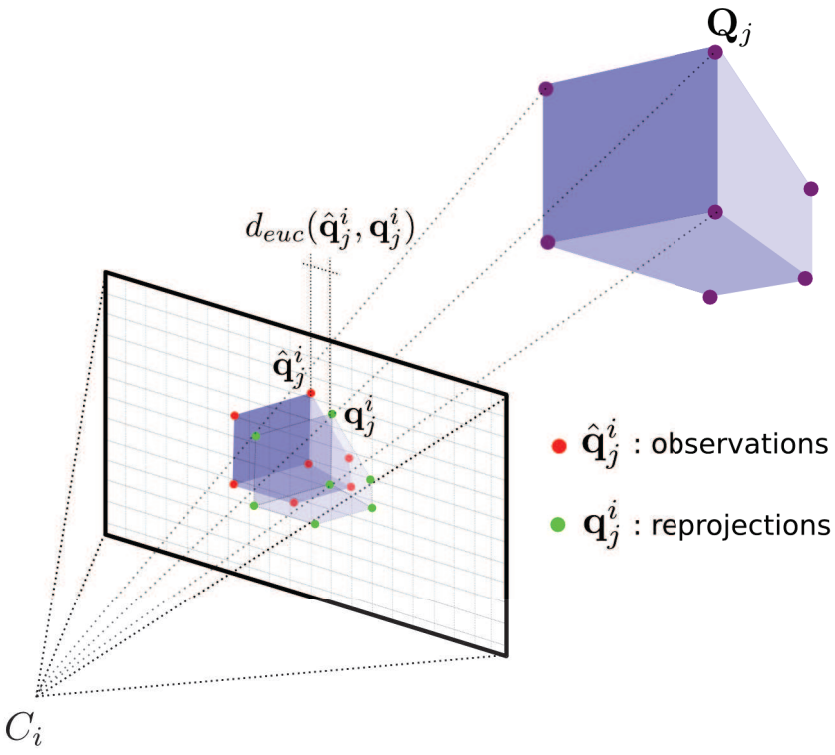


FIGURE 3.4 – Erreur de reprojection.

Il existe différentes erreurs de reprojection qui varient suivant la distance utilisée.

3.2.2.1 Erreur de reprojection géométrique

Le choix de la nature de la distance peut varier, mais il est courant d'utiliser la distance Euclidienne. On définit la fonction de coût $F^c(\mathbf{x})$ qui cumule les erreurs de reprojection géométriques par :

$$F^c(\mathbf{x}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} \nu_j^i \varepsilon_j^i(\mathbf{x}) \quad \text{avec} \quad \varepsilon_j^i(\mathbf{x}) = d(\hat{\mathbf{q}}_j^i, \mathbf{q}_j^i)^2, \tag{3.25}$$

où $\hat{\mathbf{q}}_j^i$ est l'observation détectée du point 3D $\mathbf{Q}_j$ sur l'image i et $\mathbf{q}_j^i = \mathcal{P}_i \mathbf{Q}_j$ est la reprojection du même point, calculée à partir des paramètres courants du modèle. On introduit un paramètre de visibilité ν_j^i qui nous permet de gérer les occultations de la scène ou la sélection des *inliers* : ν_j^i est à 0 lorsque le point 3D j n'est pas observé dans l'image i, ou s'il n'est pas *inlier*, et nous posons $\nu_j^i = 1$ dans les autres cas. La distance euclidienne est définie par $d(\hat{\mathbf{q}}_j^i, \mathbf{q}_j^i) = \|\hat{\mathbf{q}}_j^i - \Psi(\mathcal{P}_i \mathbf{Q}_j)\|_2$ et peut être remplacée par la distance euclidienne normalisée (ou distance de Mahalanobis, voir section 1.4.2).

L'ajustement de faisceaux géométrique est un problème de minimisation non linéaire (dont les méthodes de résolution sont décrites dans la partie 2.2.2), avec la fonction de prédiction $h_{i,j}^c(\mathbf{x})$ de l'observation $\hat{\mathbf{q}}_j^i$ définie par $h_{i,j}^c(\mathbf{x}) = \Psi\left(\mathcal{P}_i \mathbf{Q}_j\right)$ (équation (1.9)). L'ajustement de faisceaux résout le problème suivant :

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in \mathfrak{M}} F^c(\mathbf{x}). \tag{3.26}$$

Dans la suite de la thèse nous utiliserons aussi une autre mesure statistique connue sous le nom d'erreur quadratique (*RMS : Root Mean Square* en anglais), proportionnelle à la fonction de coût de l'ajustement de faisceaux et définie par :

$$\mathrm{RMS}_{2\mathrm{D}}(\mathbf{x}) = \sqrt{\frac{1}{2N_\nu} \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} \nu_j^i \varepsilon_j^i(\mathbf{x})}, \tag{3.27}$$

où N_ν est le nombre total de points d'intérêt *inlier* (donc le nombre de ν_j^i non nuls).

3.2.2.2 Erreur de reprojection algébrique

Une autre fonction de coût proposée dans la littérature [Hartley 1998] est basée sur la distance algébrique, définie dans la section 1.4.3.

La fonction de coût algébrique s'écrit :

$$F_{\mathcal{A}}^c(\mathbf{x}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} \nu_j^i \varepsilon_j^i(\mathbf{x}) \quad \text{avec} \quad \varepsilon_j^i(\mathbf{x}) = d_{\mathcal{A}}(\hat{\mathbf{q}}_j^i, \mathbf{q}_j^i)^2, \tag{3.28}$$

où $\hat{\mathbf{q}}_j^i$ est l'observation détectée du point 3D $\mathbf{Q}_j$ sur l'image i et $\mathbf{q}_j^i = \mathcal{P}_i \mathbf{Q}_j$ est la reprojection théorique du même point, calculée à partir des paramètres courants du modèle.

Il est convenu [Hartley 2003b, Chojnacki 2003] que sous une normalisation appropriée des données (une normalisation isotropique), les fonctions de coût algébriques donnent des résultats satisfaisants, eu égard au fait que cette distance ne possède pas réellement de signification géométrique ou statistique.

L'ajustement de faisceaux algébrique tente de résoudre le problème suivant :

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathfrak{M}} F_{\mathcal{A}}^{c}(\mathbf{x}). \tag{3.29}$$

L'intérêt de cette fonction de coût est qu'elle est quadratique sur les paramètres du modèle. Le problème (3.29), peut donc être résolu avec une seule itération de l'algorithme de GAUSS-NEWTON, et nous évitons ainsi les multiples itérations de l'algorithme de minimisation non-linéaire très coûteux en temps de calcul.

En pratique, cette approximation de l'erreur de reprojection entraîne un biais sur l'estimation du modèle. Le choix de l'erreur de reprojection (géométrique ou algébrique) est un compromis entre le temps de calcul accordé à l'optimisation des paramètres et la précision désirée des résultats. Aujourd'hui, la majorité des problèmes de vision par ordinateur intégrant un ajustement de faisceaux utilise l'erreur de reprojection géométrique plutôt que l'erreur de reprojection algébrique. En effet, comme nous allons voir dans la suite de ce chapitre, certaines techniques permettent aujourd'hui de résoudre le problème de l'ajustement de faisceaux (3.26) de manière relativement rapide et peu coûteuse en mémoire.

3.2.3 Résolution du problème

L'ajustement de faisceaux avec l'erreur de reprojection géométrique est un problème de minimisation non linéaire au sens des moindres carrés. Par conséquent, les techniques de minimisation présentées dans la partie 2.2.2 peuvent être employées pour résoudre l'équation (3.26). Pour simplifier les notations nous présentons ici la résolution de l'ajustement de faisceaux géométrique, dans le cas calibré. La méthode peut être aisément étendue aux différentes variantes d'ajustement de faisceaux (non-calibré, etc.).

Les techniques de résolution des moindres carrés non-linéaires opèrent de façon locale et estiment itérativement des déplacements dans l'espace des paramètres à optimiser, en résolvant l'équation (2.34) (méthode GAUSS-NEWTON) ou (2.44) (méthode LEVENBERG-MARQUARDT).

Si l'on simplifie l'équation (3.25), en ne conservant que les données robustes du problème (*inlier*) afin de ne pas calculer inutilement les reprojections des points d'intérêt *outliers*, l'équation prend alors la forme classique en moindres carrés (équation (2.11), avec $\sigma = 1$). Le vecteur d'observations $\hat{\mathbf{z}}$ est aussi réduit pour ne conserver que les coordonnées non-homogènes des N_ν points d'intérêt *inliers* Nous rappelons que les paramètres du modèle à optimiser sont définis par l'équation (3.23).

3.2.3.1 Paramétrisation des déplacements

Nous utilisons la paramétrisation locale des paramètres du modèle $\mu_k(\boldsymbol{\delta}) \in \mathfrak{M}$ autour des paramètres du modèle estimés à l'itération k de l'algorithme d'optimisation $\mathbf{x}^{(k)}$ (équation (2.18)). Le vecteur de déplacements $\boldsymbol{\delta} \in \mathbb{R}^{N_d}$ pour une itération quelconque sera constitué des éléments suivant :

$$\boldsymbol{\delta} = \begin{bmatrix} \boldsymbol{\delta}_{\mathbf{P}_1} \\ \dots \\ \boldsymbol{\delta}_{\mathbf{P}_{N_m}} \\ \boldsymbol{\delta}_{\mathbf{Q}_1} \\ \dots \\ \boldsymbol{\delta}_{\mathbf{Q}_{N_n}} \end{bmatrix} \quad \text{avec} \quad \begin{cases} \boldsymbol{\delta}_{\mathbf{P}_i} = \text{vect}(\Delta_{\mathcal{P}_i}), \text{ dans le cas non-calibré} \\ \boldsymbol{\delta}_{\mathbf{P}_i} = [\boldsymbol{\delta}_{\mathbf{e}_i} \tilde{\boldsymbol{\delta}}_{\mathbf{t}_i}], \text{ dans le cas calibré} \\ \boldsymbol{\delta}_{\mathbf{Q}_j} \text{ est un vecteur de 4 éléments (ou 3 dans le cas calibré)} \end{cases} . \tag{3.30}$$

où $\Delta_{\mathcal{P}_i}$ est une matrice 3×4 paramétrant la variation projective de la matrice de projection $\mathcal{P}_i$. Les fonctions de mises à jour incrémentales du modèle courant $\mathbf{x}^{(k)}$ vers le nouveau modèle $\mathbf{x}^{(k+1)}$ avec la paramétrisation locale $\mu_k(\boldsymbol{\delta})$ sont définis de la manière suivante. Les points 3D sont déplacés par l'équation :

$$\mathbf{Q}_j^{(k+1)} = \mathbf{Q}_j^{(k)} + \boldsymbol{\delta}_{\mathbf{Q}_j}^{(k)}. \tag{3.31}$$

En ce qui concerne les poses de la caméra, les mises à jour seront différentes suivant le type de l'ajustement de faisceaux. Dans l'ajustement de faisceaux calibré, la trajectoire de la caméra est déplacée avec la relation suivante :

$$\mathbf{t}_i^{(k+1)} = \mathbf{t}_i^{(k)} + \boldsymbol{\delta}_{\mathbf{t}_i}^{(k)}, \tag{3.32}$$

et les orientations de la caméra par (équation (2.20)) :

$$\mathcal{R}_i^{(k+1)} = \mathcal{R}_i^{(k)} \, \text{rot}_\text{e} \left(\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)} \right), \tag{3.33}$$

avec $\boldsymbol{\delta}_\mathbf{e}^\top = [\delta_\alpha \delta_\beta \delta_\gamma]$ et $\text{rot}_\text{e}(\mathbf{e})$ est la fonction (définie dans la chapitre 1) qui transforme un vecteur de 3 angles d'Euler en une matrice d'orientation. Cette représentation locale des rotations sous la forme d'angles d'Euler nous permet de linéariser localement la variété des orientations et de calculer simplement leurs dérivées partielles sur la fonction de coût.

Et dans le cas de l'ajustement de faisceaux non-calibré nous avons plus simplement :

$$\mathcal{P}_i^{(k+1)} = \mathcal{P}_i^{(k)} + \Delta_{\mathcal{P}_i}^{(k)}. \tag{3.34}$$

3.2.3.2 Construction de la matrice Jacobienne

Dans l'ajustement de faisceaux calibré, les dérivées de la fonction de coût peuvent être calculées analytiquement. Nous renvoyons le lecteur intéressé par les équations des dérivées partielles à la thèse d'ÉTIENNE MOURAGNON [Mouragnon 2007, Annexe 1]. La matrice jacobienne $\mathcal{J}_\Delta(\mathbf{x})$ des résidus $\boldsymbol{\Delta}(\mathbf{x})$ en $\mathbf{x}$ représente les relations entre les paramètres du modèle (les poses

de la caméra et les points 3D de la scène) et les observations que l'on a mesurées (positions 2D des points d'intérêt). La matrice s'écrit :

$$\mathcal{J}_{\mathbf{\Delta}}(\mathbf{x}) = \begin{pmatrix} \cdots \\ \frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{x}} \\ \cdots \end{pmatrix}_{2N_\nu \times N_p} , \tag{3.35}$$

où $\varepsilon_j^i(\mathbf{x})$ est défini dans l'équation (3.25). Une représentation de la matrice jacobienne est donnée dans la figure 3.5.

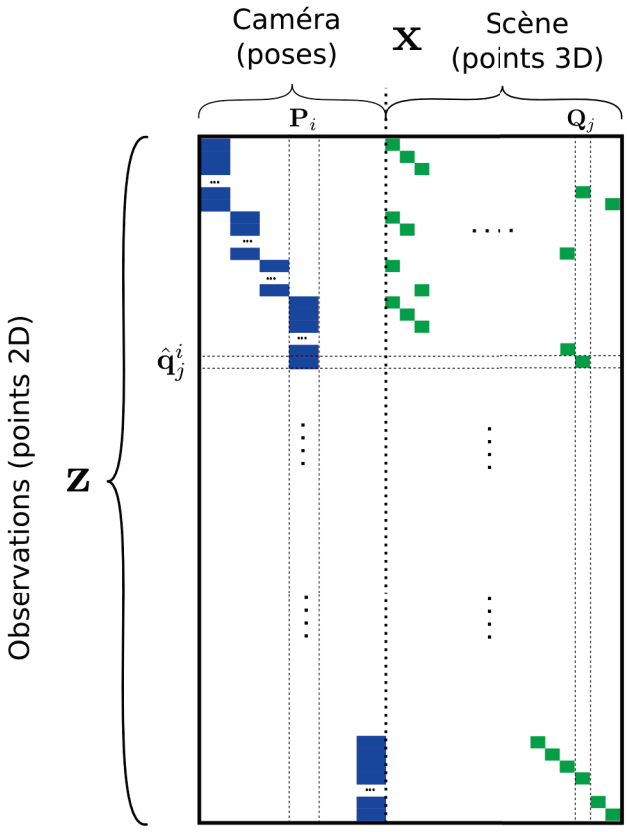


FIGURE 3.5 – Structure de la matrice jacobienne $\mathcal{J}(\mathbf{x})$ sur un problème de SLAM monoculaire. Les éléments colorés sont constitués de valeur non nulles.

La matrice jacobienne du système est souvent une matrice creuse. En effet, chaque ligne représente (une coordonnée de) la projection d'un seul point 3D dans une seule image par conséquent, chaque ligne ne possède que deux blocs non nuls. La matrice jacobienne possède donc une structure par blocs (de taille 2×6 pour la partie caméra et 2×3 dans la partie scène, dans le cas d'un ajustement de faisceaux calibré), avec de nombreux blocs ne contenant que des valeurs nulles.

Pour réduire la quantité de mémoire nécessaire à la résolution du problème et accélérer le processus d'inversion (voir équation (2.15)), la matrice jacobienne peut ne pas être stockée. L'idée est alors de ne calculer et conserver en mémoire que les variables de l'équation (2.44), c'est-à-dire la matrice approximant la matrice hessienne $\mathcal{H} \approx \mathcal{B} = {\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta$ (présentée en figure 3.6) et le gradient du système $\boldsymbol{\nabla}(\mathbf{x}) = \mathcal{J}(\mathbf{x})^\top \boldsymbol{\Delta}(\mathbf{x})$. Dès lors, au lieu de stocker la matrice jacobienne, nous calculons directement ces deux matrices à partir de la matrice jacobienne.

Nous pouvons ensuite procéder à la minimisation de la fonction de coût par la technique de LEVENBERG-MARQUARDT (voir section 2.2.2.4). À partir d'une solution initiale $\mathbf{x}_0$ du problème, nous réalisons plusieurs itérations de l'algorithme, en résolvant l'équation (2.44), jusqu'à convergence de la méthode.

La structure particulière de la matrice hessienne approximée par l'approximation de GAUSS-NEWTON $\mathcal{H} \approx {\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta$ (lorsque les variances des mesures sont omises) rend possible la résolution de l'équation (2.44) de manière très rapide, par le complément de SCHUR [Triggs 2000].

3.2.3.3 Résolution du problème par le complément de SCHUR

Tout comme la matrice jacobienne, la matrice symétrique $\mathcal{B}$ possède une structure creuse et peut être stockée par blocs [Hartley 2003b], comme le montre la figure 3.6. En effet, d'une part, les occultations fréquentes de la scène font que nous n'observons pas tous les points 3D dans toutes les images. Et d'autre part, les poses des caméras peuvent être considérées comme indépendantes les unes des autres, tout comme les points 3D.

Nous pouvons dès lors décomposer la matrice $\mathcal{B}$ de la manière suivante :

$$\mathcal{B} = \begin{pmatrix} \mathcal{U} & \mathcal{W}^\top \\ \mathcal{W} & \mathcal{V} \end{pmatrix}, \tag{3.36}$$

où $\mathcal{U}$ est une matrice carré (de taille $(N_p^c)^2$, ou $6N_m \times 6N_m$ dans le cas calibré) contenant des blocs diagonaux (6×6) représentant les relations entre les observations 2D dans une image i et les paramètres de la caméra associés. Le bloc i de la matrice $\mathcal{U}$ est définie par :

$$\mathcal{U}^i = \sum_j \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{P}_i} \right)^\top \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{P}_i} \right), \tag{3.37}$$

où $\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{P}_i}$ est la matrice de taille 2×6 qui contient les dérivées de $F^c(\mathbf{x})$ par rapport aux paramètres extrinsèques de la caméra c_i.

$\mathcal{V}$ est une matrice carré (de taille $(N_p^Q)^2$, ou $3N_n \times 3N_n$ dans le cas calibré) diagonale par bloc (3×3, ou 4×4 en non calibré). Chaque bloc représente les relations entre les coordonnées d'un point 3D Q_j et les observations du point dans les différentes images. La formule du bloc j est :

$$\mathcal{V}_j = \sum_j \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{Q}_j} \right)^\top \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{Q}_j} \right), \tag{3.38}$$

où $\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{Q}_j}$ est la matrice de taille 2×6 qui contient les dérivées de $F^c(\mathbf{x})$ par rapport aux paramètres (coordonnées 3D) du point Q_j. La matrice $\mathcal{V}$ est facilement inversible étant donné sa structure

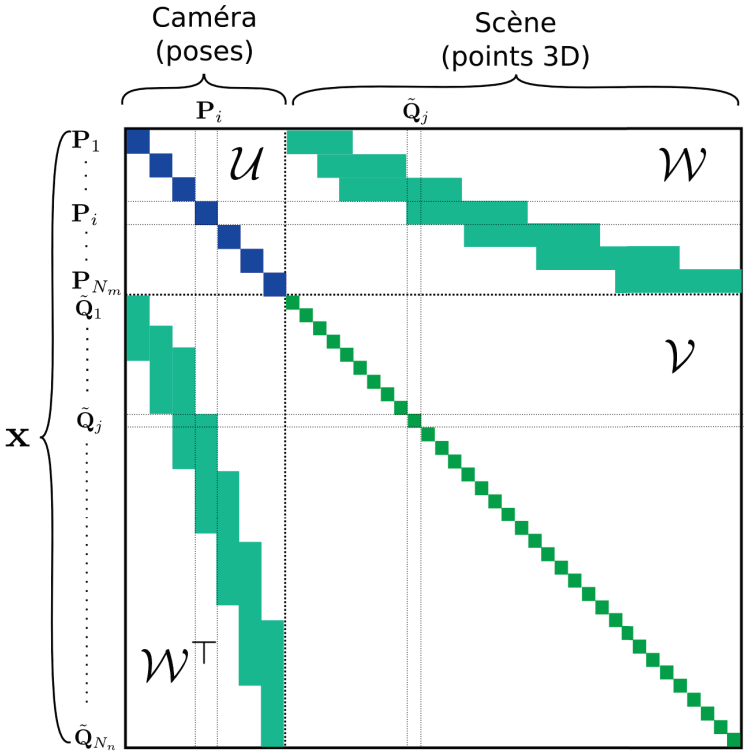


FIGURE 3.6 – Structure de la matrice hessienne approximée $\mathcal{H} \approx \mathcal{B} = {\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta$.

et la méthode du complément de SCHUR va utiliser cette propriété pour résoudre rapidement le système.

Enfin, la matrice $\mathcal{W}$ (de taille $N_p^c \times N_p^Q$, ou $6N_m \times 3N_n$ dans le cas calibré) exprime les inter-corrélations entre les paramètres des points 3D et les paramètres des poses de la caméra. Le bloc $\mathcal{W}_j^i$ de cette matrice s'écrit :

$$\mathcal{W}_j^i = \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{Q}_j}\right)^\top \left(\frac{\partial \varepsilon_j^i(\mathbf{x})}{\partial \mathbf{P}_i}\right). \tag{3.39}$$

Nous pouvons décomposer l'équation (2.34) en :

$$\begin{pmatrix} \mathcal{U} & \mathcal{W}^\top \\ \mathcal{W} & \mathcal{V} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_\mathcal{P} \\ \boldsymbol{\delta}_Q \end{pmatrix} = \begin{pmatrix} \boldsymbol{\nabla}_\mathcal{P} \\ \boldsymbol{\nabla}_Q \end{pmatrix}. \tag{3.40}$$

Le principe du complément de SCHUR est alors de décomposer la résolution de l'équation (2.34) en deux étapes. Nous calculons en premier lieu le vecteur $\boldsymbol{\delta}_\mathcal{P}$ des déplacements des paramètres des poses de la caméra en résolvant le système linéaire suivant :

$$(\mathcal{U} - \mathcal{W}^\top \mathcal{V}^{-1} \mathcal{W}) \boldsymbol{\delta}_\mathcal{P} = \boldsymbol{\nabla}_\mathcal{P} - \mathcal{W}^\top \mathcal{V}^{-1} \boldsymbol{\nabla}_Q. \tag{3.41}$$

Ce système a été obtenu en multipliant l'équation (3.40) à gauche par la matrice $\begin{pmatrix} \mathcal{I} & -\mathcal{W}^\top \mathcal{V}^{-1} \\ 0 & \mathcal{I} \end{pmatrix}$:

$$\begin{pmatrix} \mathcal{U} - \mathcal{W}^\top \mathcal{V}^{-1} \mathcal{W} & 0 \\ \mathcal{W} & \mathcal{V} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_\mathcal{P} \\ \boldsymbol{\delta}_Q \end{pmatrix} = \begin{pmatrix} \boldsymbol{\nabla}_\mathcal{P} - \mathcal{W}^\top \mathcal{V}^{-1} \boldsymbol{\nabla}_Q \\ \boldsymbol{\nabla}_Q \end{pmatrix}. \tag{3.42}$$

Ensuite, nous calculons directement les déplacements $\boldsymbol{\delta}_Q$ des points 3D à partir de $\boldsymbol{\delta}_\mathcal{P}$.

L'ajustement de faisceaux utilisant le complément de SCHUR s'avère être une technique très rapide et efficace puisque nous inversons seulement la matrice $\mathcal{V}$, qui est diagonale par bloc 3×3. Nous pouvons montrer que la complexité d'une itération d'un ajustement de faisceaux utilisant ce procédé est $O(N_\nu + N_m^3)$.

Nous avons ici présenté la résolution d'un ajustement de faisceaux par la technique de GAUSS-NEWTON utilisant le complément de SCHUR, mais la méthode peut tout aussi bien être employée dans une méthode de LEVENBERG-MARQUARDT. En effet, il suffit d'ajouter (ou multiplier suivant les variantes) la matrice de régularisation ($\lambda \mathcal{D}$) à la matrice $\mathcal{B}$.

3.2.4 Ajustement par alternance

L'ajustment par alternance, ou *alternation* en anglais, est une approche simplifiée et souvent moins précise que l'ajustement de faisceaux. Le principe est de décomposer les variables à optimiser en plusieurs groupes et de n'optimiser qu'un seul groupe à la fois [Morris 1998, Wang 2001, Hartley 2003a] en ignorant l'interdépendance entre les groupes. Chaque groupe de variables sera alors optimisé à tour de rôle en fixant les autres variables. Un exemple d'alternance est la technique de resection-intersection [Bartoli 2002, Liu 2008], dans laquelle on optimise alternativement la ou les poses de la caméra, en fixant les points 3D de la scène, puis on optimise la structure de la scène en fixant les poses de la caméra. On réitère ensuite le processus jusqu'à convergence.

Ces sous-problèmes étant considérés comme indépendants, seul le bloc diagonal de la matrice hessienne est conservée et la méthode revient à réaliser plusieurs optimisation de premier ordre. Par exemple, pour la technique de resection-intersection, on inverse alternativement les matrices creuses $\mathcal{U}$ et $\mathcal{V}$.

L'alternance se veut être une technique très rapide puisqu'elle ne réalise que des optimisations de premier ordre. Mais elle peut avoir le désavantage de converger lentement aux environs du minimum du fait de l'approximation de la matrice hessienne. On pourra noter qu'avec l'utilisation d'une technique de recherche linéaire, la convergence en sera améliorée [Triggs 2000], sans pour autant être aussi généralement rapide qu'une méthode du second ordre (voir section 2.2.2).

Cette technique s'avère être en pratique d'une efficacité réduite sur des problèmes de grande dimensions, mais elle peut être utile sur des problèmes réduits et très connectés (beaucoup d'observations) [Triggs 2000].

3.3 *Structure-from-Motion*

Depuis quelques décennies les travaux de recherche dans le domaine de la vision par ordinateur se sont fortement orientés vers les problèmes de reconstruction 3D.

Dans de nombreux problèmes liés à l'image ou à la vidéo, le calcul de la structure 3D de la scène filmée ainsi que de la localisation de la caméra est essentiel. Ce calcul, effectué à partir des images 2D, est connu sous le nom de reconstruction par le mouvement (*Structure-from-Motion*). Une introduction au problème de la reconstruction 3D à partir d'une séquence d'images est donné par POLLEFEYS *et al.*, voir [Pollefeys 2004].

On peut classer les algorithmes de *Structure-from-Motion* en 2 grandes familles : le *batch Structure-from-Motion* et le *Structure-from-Motion* incrémental ou séquentiel. Ces familles diffèrent principalement par leur manière de traiter les images et dépendent du schéma d'obtention des images. Avec les méthodes de *batch Structure-from-Motion* [Hartley 1999, Nistér 2004b, Martinec 2008], l'ensemble des images sont traitées simultanément. Une variante consiste à traiter les images par sous ensembles et de les fusionner ensuite de manière hiérarchique afin de reconstruire l'intégralité de la scène. Ce sont les méthodes dites de *Structure-from-Motion* hiérarchique [Fitzgibbon 1998, Shum 1999, Nistér 2000, Farenzena 2009, Ni 2010]. La seconde catégorie regroupe les algorithmes de *Structure-from-Motion* incrémental [Royer 2006, Mouragnon 2007, Engels 2006, Klein 2007], aussi connus sous le nom de SLAM (*Simultaneous Localization And Mapping*) dans la communauté robotique. Dans cette dernière catégorie, les images sont traitées les unes après les autres, ce qui rend la méthode utilisable dans un contexte temps réel où l'on acquière dynamiquement les images provenant d'une caméra.

Ces différentes méthodes se composent de deux principales étapes : d'une part il s'agit d'estimer le mouvement de la caméra (ou les matrices de projection) et la structure de la scène par un processus algébrique, puis de raffiner ces estimations par un algorithme non linéaire, l'ajustement de faisceaux.

Dans cette thèse nous avons employé ces deux types de *Structure-from-Motion* : le *batch Structure-from-Motion* et le *Structure-from-Motion* incrémental. Nous présentons succinctement ces deux familles ainsi que les algorithmes spécifiquement employés. Les différents détails d'implémentation de ces techniques ne sont pas mentionnés, nous renvoyons le lecteur intéressé aux publications indiquées.

3.3.1 *Batch Structure-from-Motion*

3.3.1.1 Présentation

Dans la problématique de localisation de caméras et de reconstruction de scène, la disponibilité des images va conditionner le choix de la technique de *Structure-from-Motion* à employer. Dans le cas hors ligne (sans la contrainte temps réel) et lorsque toutes les images sont disponibles, alors le *batch Structure-from-Motion* peut être appliqué.

Le principe général du *batch Structure-from-Motion* est de commencer par estimer une reconstruction projective de la scène à l'aide de la factorisation perspective (décrite dans la sec-

tion 3.1.5), puis de rectifier celle-ci afin de la rendre métrique [Hartley 2003b]. Ce dernier problème est connu sous le nom d'autocalibrage. Une synthèse des méthodes de *batch Structure-from-Motion* projective est présentée dans la thèse de MARTINEC [Martinec 2008]. L'algorithme 4 résume les différentes étapes de la méthode.

Algorithme 4 : `BatchSfm` : Algorithme de *batch Structure-from-Motion* par factorisation

- Données : Les observations 2D $\hat{\mathbf{q}}_i$
- Sorties : Les matrices de projection $\mathcal{P}_i$ et les points 3D $\mathbf{Q}_j$ de la scène

```
// Factorisation perspective
```
Normaliser les données $\hat{\mathbf{q}}'_i = \mathcal{N}_i \hat{\mathbf{q}}_i$ [équation (4.3)] ;
Estimer les matrices fondamentales $\mathcal{F}_{ik}$ et leurs épipoles $\mathbf{e}_{ik}$ [équation (3.8)];
Construire la matrice de mesure $\mathcal{M}$ [équation (3.19)];
Estimer les profondeurs projectives λ^i_j [section 3.1.5.2];
Pondérer la matrice de mesure $\mathcal{M}$ avec ces nouvelles profondeurs;
si *Il n'y a aucune occultation de scène* **alors**
  `// Factorisation perspective par SVD [section 3.1.5.3]`
  Décomposition SVD de $\mathcal{M}$ pour retrouver les matrices de projections $\mathcal{P}_i$ et les coordonnées des points 3D $\mathbf{Q}_j$ de la scène;
sinon
  `// Factorisation perspective avec contraintes [section 3.1.5.4]`
  tant que *Il reste des matrices de projection non estimées* **faire**
    Rechercher des sous-matrices complètes;
    Appliquer les contraintes (CBC, ...);
    Résoudre le système pour retrouver les matrices de projection;
  Trianguler tous les points 3D;
si *on désire une reconstruction métrique* **alors**
  `// Rectification métrique [section 3.3.1.2]`
  si *la matrice de calibrage est inconnue* **alors**
    Autocalibrer et retrouver la matrice $\mathcal{K}$ et le plan à l'infini π_∞;
  sinon
    Rechercher le plan à l'infini π_∞;
  Appliquer la rectification;
```
// Optimisation non linéaire
```
Ajustement de faisceaux global [section 3.2];

3.3.1.2 Rectification métrique

Rappelons que la méthode de factorisation ne reconstruit la scène qu'à une homographie près (équation (3.22)). Cette homographie, que l'on notera $\mathcal{H}$, représente la rectification à apporter pour retrouver la vraie scène métrique, définie à une similitude près (rotation, translation et échelle). Le but de la rectification métrique est donc d'estimer cette homographie, puis d'ap-

pliquer celle-ci sur la scène et les matrices de projection par :

$$\mathcal{P}_i^{'} = \mathcal{P}_i\mathcal{H} \quad \text{et} \quad \mathbf{Q}_j^{'} = \mathcal{H}^{-1}\mathbf{Q}_j. \tag{3.43}$$

Notons que si l'on impose $\mathcal{P}_1 = (\mathcal{I} \mid \mathbf{0})$, alors l'homographie $\mathcal{H}$ peut se décomposer ainsi :

$$\mathcal{H} = \begin{pmatrix} \mathcal{K} & \mathbf{0} \\ -\mathbf{p}^\top\mathcal{K} & 1 \end{pmatrix}, \tag{3.44}$$

où $\pi_\infty = [\mathbf{p}^\top 1]^\top$ est le vecteur contenant les coordonnées du plan à l'infini [Hartley 2003b, p. 458].

Grâce à cette décomposition de la matrice $\mathcal{H}$, le problème de la rectification métrique revient donc à estimer deux choses : les paramètres intrinsèques de la caméra $\mathcal{K}$ et les coordonnées du plan à l'infini π_∞. Il existe plusieurs manières de résoudre ce problème connu aussi sous le nom d'autocalibrage [Hartley 2003b, p. 458]. Lorsque les paramètres de calibrage de la caméra (matrice $\mathcal{K}$) ne sont pas connus, les méthodes d'autocalibrage sont nécessaires. Elles emploient en général des contraintes linéaires (sous certaines hypothèses) issues de l'image de la conique absolue duale (DIAC) : $\omega^* = \mathcal{K}\mathcal{K}^\top$, projection de la quadrique absolue duale $\mathcal{Q}_\infty^\star$ telle que $\omega^* = \mathcal{P}_i\mathcal{Q}_\infty^\star\mathcal{P}_i^\top$, pour estimer les éléments de la matrice de calibrage. Dans l'approche stratifiée, on s'intéresse d'abord à estimer les coordonnées du plan à l'infini et ainsi obtenir une reconstruction quasi-affine, puis on estime la matrice de calibrage.

Lorsque les paramètres intrinsèques de la caméra sont connus, ce qui est le cas dans une majorité de nos expérimentations, seuls les coordonnées du plan à l'infini demeurent à estimer. Pour cela, il est possible d'utiliser des contraintes sur ses coordonnées, issues de la quadrique absolue. Voir notamment les travaux de HARTLEY [Hartley 1999] et NISTER [Nistér 2004b].

3.3.2 *Structure-from-Motion* incrémental

Dans cette section nous nous intéressons au processus basé sur les algorithmes de *Structure-from-Motion*, qui permettent de localiser un système en temps réel à l'aide d'une caméra monoculaire.

3.3.2.1 Présentation

Le *Structure-from-Motion* incrémental (ou séquentiel) est un outil de localisation et cartographie simultanée (SLAM) monoculaire dans lequel les images sont traitées séquentiellement, l'une après l'autre. Il existe différentes versions de SLAM monoculaire, utilisant notamment un filtre de KALMAN [Davison 2003], ou un ajustement de faisceaux [Mouragnon 2007, Engels 2006, Klein 2007]. Dans cette thèse, nous avons employé la technique de SLAM monoculaire temps réel proposée par MOURAGNON *et al.* dans [Mouragnon 2006], dont les grandes étapes sont résumées dans l'algorithme 5.

Le principe général de cette méthode (schématisée dans la figure 3.7) est d'une part de localiser la caméra au cours du temps, et d'autre part de reconstruire la scène de manière incrémentale, en s'appuyant sur les reconstructions précédentes. Ce processus est possible grâce à

un traitement 2D consistant à suivre (détecter, décrire et apparier) des points d'intérêt dans les images. Un algorithme d'optimisation est régulièrement appliqué afin de réduire les erreurs cumulées.

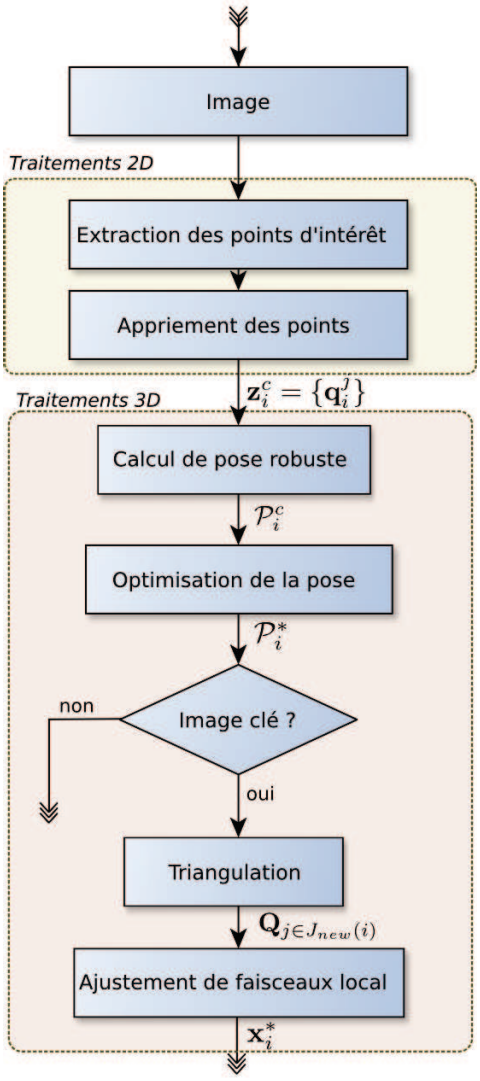


FIGURE 3.7 – Schéma de fonctionnement du SLAM incrémental de MOURAGON *et al.* [Mouragnon 2006].

Sans entrer dans les détails d'implémentation, nous allons détailler ci-après quelques unes des étapes spécifiques de la méthode, introduites par MOURAGNON *et al.* dans [Mouragnon 2006].

3.3.2.2 Sélection des images clés

Le déplacement d'une caméra vidéo entre deux vues consécutives étant généralement trop limité pour pouvoir appliquer les algorithmes de reconstruction avec précision, le SLAM monoculaire de MOURAGNON *et al.* [Mouragnon 2006] propose de ne reconstruire la scène que sur cer-

taines images, dites images clés. Pour cela, les auteurs définissent un critère permettant de définir si une image est une image clé ou non. Ce critère, essentiellement 2D, est notamment basé sur le nombre appariements 2D avec l'image clé précédente. Lorsque le nombre d'appariements 2D entre l'image et la dernière image clé est inférieur à un seuil M ($M = 400$ dans notre cas), alors l'image courante est considérée comme une image clé.

3.3.2.3 Reconstruction et localisation initiales

Au début de la séquence d'images, l'algorithme de *Structure-from-Motion* incrémental n'a pas encore d'information sur la localisation de la caméra ni sur la structure de la scène. Mais avec l'ensemble des points d'intérêt détectés et appariés dans les 3 premières images clés, il est possible de déterminer de manière robuste la trajectoire de la caméra et la structure initiale de la scène. Pour cela, la première pose $\mathcal{T}_1$ est fixée à l'origine. Ensuite, le déplacement entre les caméras $\mathcal{T}_1$ et $\mathcal{T}_2$ est calculé de manière robuste par la méthode décrite dans la section 3.1.4. Les points 3D vus dans ces deux poses sont triangulés (section 3.1.2) et la dernière pose $\mathcal{T}_3$ est calculée par resection (section 3.1.3).

Ce premier modèle (les trois poses et la scène) est ensuite raffiné au moyen d'un ajustement de faisceaux global (section 3.2), c'est-à-dire sur l'ensemble des données (*inlier*).

3.3.2.4 Reconstruction et localisation incrémentales

Ensuite, le déplacement de chaque nouvelle image clé détectée est identifié grâce aux points 3D précédemment reconstruits, par la méthode de resection (section 3.1.3). De nouveaux points 3D non encore reconstruits et vus dans les trois dernières images clés sont estimés par triangulation (section 3.1.2) à l'aide du nouveau déplacement inter-caméras clés calculé précédemment, c'est l'intersection.

Un raffinement local est ensuite périodiquement appliqué sur les dernières poses de la caméra et sur la scène pour réduire les erreurs cumulées.

Algorithme 5 : `SlamSfM` : Algorithme de *Structure-from-Motion* incrémental (SLAM monoculaire).

- Entrées : Une nouvelle image

`// Traitements 2D`
Détecter les $N = 200$ meilleurs points 2D [section 3.1.1.1] ;
Calculer les descripteurs de ces points d'intérêt [section 3.1.1.2] ;
Apparier rapidement ces points avec les 2 dernières images clé [section 3.1.1.3] ;
`// Traitements 3D`
Détecter si l'image est une image clé [section 3.3.2.2] ;
si *c'est une image clé* **alors**
  si *la scène n'est pas initialisée* **alors** `// Processus d'initialisation`
    si *il s'agit de la 3ème image clé détectée* **alors**
      `// Calcul des poses initiales`
      Estimer les 3 poses initiales à partir des correspondances 2D [section 3.1.4];
      `// Intersection (Mapping)`
      Trianguler les points 3D [section 3.1.2];
      `// Optimisation non linéaire`
      Ajustement de faisceaux global [section 3.2];
    sinon
      Consever la nouvelle image clé;
  sinon `// Processus incrémental`
    `// Resection`
    Estimer le déplacement inter-images clés avec la scène reconstruite [section 3.1.3];
    `// Intersection (Mapping)`
    Trianguler les nouveaux points 3D [section 3.1.2];
    `// Optimisation non linéaire`
    Ajustement de faisceaux local sur les $N_{LBA} = 3$ dernières caméras
    [section 3.3.2.5];

3.3.2.5 Ajustement de faisceaux local

Le principe de l'ajustement de faisceaux local (ou glissant) décrit dans [Mouragnon 2006] consiste à n'optimiser que les N_{LBA} dernières caméras. En effet, la complexité de l'ajustement de faisceaux par la méthode du complément de SCHUR est cubique par rapport au nombre de ses paramètres. Si l'on souhaite utiliser l'ajustement de faisceaux dans une application temps-réel, il est nécessaire de limiter les paramètres à optimiser. Les auteurs de [Mouragnon 2006] ont montré que l'optimisation locale des $N_{LBA} = 3$ dernières poses de la caméra était suffisante pour localiser et cartographier relativement précisément un environnement inconnu. Une illustration de l'ajustement de faisceaux local est proposée en figure 3.8.

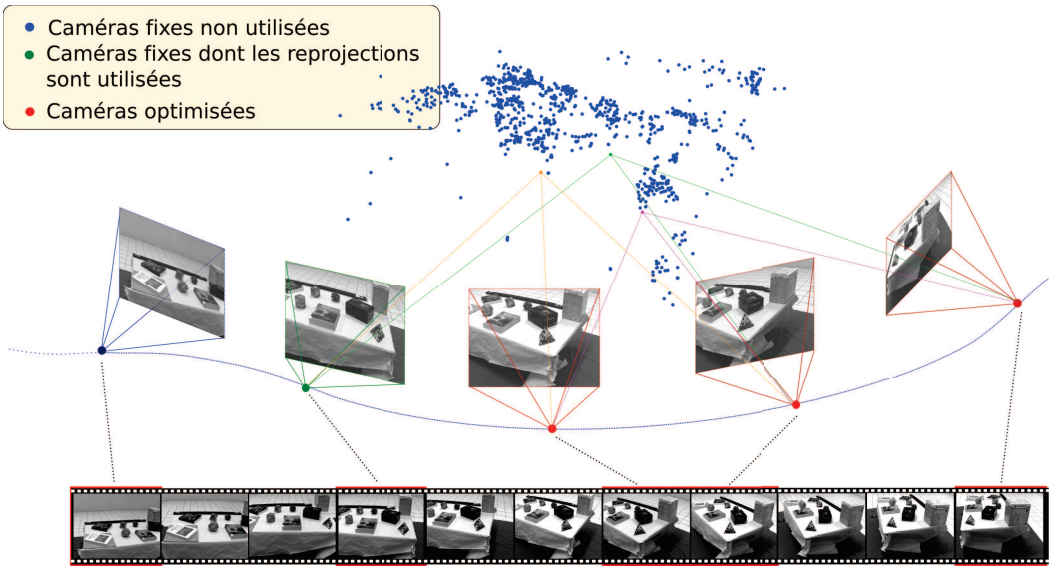


FIGURE 3.8 – Ajustement de faisceaux local. Les caméras et les points rouges sont optimisées, les erreurs de reprojection des caméras vertes sont intégrées dans la fonction de coût et les caméras bleues ne sont pas prises en compte.

La sélection des points 3D à optimiser peut varier suivant les méthodes. Dans cette thèse, nous optimisons l'ensemble des points 3D vus dans les N_{LBA} dernières caméras. La fonction de coût de l'ajustement de faisceaux local comprend les erreurs de reprojection de ces points dans toutes les images où ceux-ci apparaissent. Notons que les points 3D vus dans moins de 3 caméras ou ayant une erreur de reprojection supérieure à un seuil de 4 pixels (*outlier*) ne sont pas optimisés. Soit $\mathbf{s}_i$ cet ensemble de points, tel que $\mathbf{s}_i = [\mathbf{Q}_{j|\nu_j^{i'}=1 \ \forall i' \in [i-N_{LBA}+1,i] \ \forall j \in [1,N_n]}]$, où i est la nouvelle image clé.

Après l'estimation par l'algorithme de resection de la nouvelle caméra clé i, et la triangulation des nouveaux points 3D, nous procédons à un ajustement de faisceaux local, dont le vecteur de paramètres à optimiser est

$$\mathbf{x}_i = [\mathbf{P}_{i-(N_{LBA}-1)}\mathbf{P}_{i-(N_{LBA}-2)} \dots \mathbf{P}_i \mathbf{s}_i], \tag{3.45}$$

et la fonction objectif est :

$$F^{LBA}(\mathbf{x}_i) = \sum_{i'=1}^{N_m} \sum_{j|\mathbf{Q}_j \in \mathbf{s}_i} \nu_j^{i'} \varepsilon_j^{i'}(\mathbf{x}_i). \tag{3.46}$$

La résolution du problème de l'ajustement de faisceaux local (minimisation de la fonction objectif (3.46)) est ensuite effectuée par le même procédé que pour l'ajustement de faisceaux global, décrit dans la section 3.2, mais en un temps beaucoup plus faible étant donné le nombre restreint de paramètres simultanément optimisés.

3.3.2.6 Limites de la méthode

Les méthodes de SLAM monoculaire actuelles présentent certaines limites. Tout d'abord, comme nous l'avons précisé, la reconstruction de la scène ainsi que la trajectoire de la caméra ne sont estimées qu'à un facteur d'échelle près. Les informations ne sont pas non plus géoréférencées (dans un repère global) mais sont exprimées dans un repère local que l'on a fixé. L'apport d'*a priori* sur la scène ou d'informations extérieures précises sur la trajectoire de la caméra provenant par exemple d'autres capteurs peut résoudre ces problèmes.

De plus, le SLAM monoculaire étant une technique de localisation et reconstruction incrémentale, il est sensible à l'accumulation d'erreur. En effet, les processus de détection et de suivi de points d'intérêt n'étant pas parfaits, les erreurs effectuées sur les mesures 2D sont propagées aux estimations 3D (localisation et reconstruction) et une dérive apparaît sur la trajectoire de la caméra. Ce problème est d'autant plus présent lorsque les trajectoires sont longues (quelques kilomètres), mais dépend aussi de la rigidité de la scène filmée, de la rapidité de mouvement de la caméra etc.

Une autre dérive est également présente dans le SLAM monoculaire, c'est la dérive du facteur d'échelle. Le facteur d'échelle peut en effet évoluer au cours du temps et l'on se retrouve alors avec différentes échelles locales sur une même séquence. Les causes de cette dérive ne sont aujourd'hui encore pas bien identifiées. Il pourrait s'agir d'une mauvaise répartition des points d'intérêt dans l'image, ou une mauvaise estimation de la profondeur des points 3D.

3.4 Conclusion

Nous venons de présenter quelques algorithmes de vision par ordinateur permettant de localiser une caméra par son mouvement et de reconstruire la scène rigide observée. Dans les chapitres suivant nous allons employer l'ensemble de ces techniques et adapter ou améliorer certaines d'entre-elles. Ces techniques serviront à concevoir d'une part, une application de reconstruction 3D hors ligne avec un processus d'ajustement de faisceaux accéléré et d'autre part, une application de localisation en temps réel par un processus de *Structure-from-Motion* séquentiel contraint par des informations provenant de capteurs inertiels ou odométriques.

CHAPITRE 4

Recherche linéaire pour la reconstruction 3D

Dans ce chapitre nous présentons une méthode permettant d'améliorer la rapidité de convergence des optimisations non-linéaires dans les méthodes de Structure-from-Motion itératives. Après avoir présenté les différentes méthodes d'optimisation de l'état de l'art, nous proposons une nouvelle technique de recherche linéaire. Nous détaillons ensuite comment cette technique peut être employée dans un algorithme d'ajustement de faisceaux, dans le cadre d'une application de batch Structure-from-Motion. Cette méthode ainsi que ses variantes ont donné lieu à plusieurs publications : CORESA [Michot 2009c], BMVC [Michot 2009a] et ORASIS [Michot 2009b].

In this chapter we present a method that improves the convergence speed of nonlinear optimizations in the methods of Structure-from-Motion. After presenting the different optimization methods of the prior art, we propose a new technique of line search. We then detail how this technique can be used in a calibrated or uncalibrated bundle adjustment, through an application of batch Structure-from-Motion. This method and its variants have been published in the following conferences : CORESA [Michot 2009c], BMVC [Michot 2009a] and ORASIS [Michot 2009b].

4.1 État de l'art

4.1.1 Problématique

De nombreux problèmes de vision par ordinateur ne sont pas résolvables linéairement et il alors courant de faire appel à un algorithme d'optimisation qui minimise un critère non-linéaire par rapport aux paramètres estimés. Ce critère est généralement composé de distances 2D, comme par exemple dans le problème d'estimation de la matrice fondamentale où l'on cherche à minimiser les distances entre des point 2D et les lignes épipolaires associées. D'autres problèmes de *Structure-from-Motion* minimisent aussi un critère basé composé de distances 2D (ou 3D), mentionnons notamment l'estimation de la matrice essentielle, l'optimisation de la structure de la scène ou de la trajectoire de la caméra, voire les deux simultanément, ce que l'on appelle l'ajustement de faisceaux.

Ces problèmes de moindres carrés non-linéaires sont, comparativement aux algorithmes d'estimation linéaires, complexes et extrêmement coûteux en temps de calcul. En effet, comme nous l'évoquons dans la section 2.2.2, les méthodes de résolution de ces problèmes sont itératifs, et nécessitent en général plusieurs itérations pour converger vers une solution stationnaire. De plus il n'existe aucune garantie sur le fait que la solution trouvée soit la solution globale du problème.

Le temps de calcul total nécessaire à l'algorithme d'optimisation pour converger vers une solution stationnaire dépend principalement de deux choses : la complexité du problème et la qualité de la solution initiale. La complexité du problème va déterminer le nombre d'opérations qui seront à réaliser par l'ordinateur à chaque itération de l'algorithme. Si l'on prend l'exemple de l'ajustement de faisceaux (voir section 3.2), celui-ci a une complexité par itération (résolution de l'étape de GAUSS-NEWTON) assez importante puisqu'elle est en $\mathcal{O}(N_p^{c3})$, où N_p^c est le nombre de paramètres des caméras à optimiser (ou $\mathcal{O}({N_p}^3)$ dans sa forme dense, sans le complément de SHUR) [Triggs 2000]. La qualité de la solution initiale va quant-à-elle indiquer le nombre d'itérations à réaliser. Plus l'estimation initiale est imprécise, et plus l'algorithme d'optimisation doit réaliser d'itérations pour converger vers une solution efficace. De plus, l'estimation est dans ces conditions beaucoup plus sensibles aux minima locaux de la fonction objectif.

Ainsi, de nombreux travaux de recherche ont été entrepris pour améliorer la convergence des algorithmes de moindres carrés non-linéaires, aussi bien dans le cadre général que dans les applications de vision par ordinateur.

4.1.2 Méthodes existantes et limites

Nous nous intéressons ici aux algorithmes employés afin de résoudre les problèmes d'optimisation non-linéaire en vision par ordinateur. Les principes généraux de ces techniques sont présentés dans la section 2.2.2 et nous mentionnons ici leurs utilisations dans l'état de l'art.

Ces algorithmes peuvent être classées en deux groupes : les méthodes qui s'intéressent à faire converger l'optimisation plus efficacement, et celles qui réduisent le nombre de paramètres de l'optimisation et la complexité du problème. Dans cette dernière catégorie nous présentons les travaux effectués principalement dans le cadre de l'ajustement de faisceaux.

4.1.2.1 Différentes méthodes d'optimisation

Dans le domaine de la vision par ordinateur, les applications de reconstruction et localisation par l'image nécessitent l'usage d'un algorithme d'optimisation non-linéaire, pour raffiner la matrice fondamentale, la trajectoire d'une caméra, les poses de plusieurs caméras regardant une même scène, la reconstruction d'une scène et les deux éléments (localisation et reconstruction) dans un ajustement de faisceaux.

Les techniques des moindres carrés non-linéaires adoptées pour résoudre le problème varient principalement suivant le type de fonction objectif (sa non-linéarité), du nombre de variables à optimiser simultanément (10, 1000, 10000 ?) et la qualité de la solution initiale (distance par rapport à l'optimum aux sens des moindres carrés).

Optimisations par GAUSS-NEWTON et variantes. Habituellement, les techniques de minimisation de type GAUSS-NEWTON (section 2.2.2.2) sont employées afin de bénéficier de la convergence quadratique de la méthode. Lorsque les solutions initiales, délivrées par exemple par une approximation linéaire du problème, peuvent être estimées assez précisément, la méthode de GAUSS-NEWTON seule suffit généralement à réaliser l'optimisation et à converger vers la solution optimale du problème.

Mais dans la majorité des cas, il est préférable de choisir une variante de cette technique, comme LEVENBERG-MARQUARDT, décrit en section 2.2.2.4. Cette technique est reconnue pour être l'une des techniques d'optimisation non-linéaire les plus efficace, même avec une initialisation assez dégradée. Une grand majorité des problèmes non-linéaires en vision par ordinateur, que ce soit pour optimiser la matrice fondamentale [Luong 1995, Hartley 1995, Zhang 2001, Bartoli 2004] que dans l'ajustement de faisceaux [Shum 1999, Triggs 2000, Hartley 2003b, Buchanan 2005, Hung 2006, Mouragnon 2006, Klein 2007, Ni 2007, Snavely 2008b, Lourakis 2009, Strecha 2010], sont résolus par cette technique. Celle-ci est d'ailleurs l'implémentation choisie par LOURAKIS *et al.* dans SBA (*Sparse Bundle Adjustment*) [Lourakis 2009]. La méthode de LEVENBERG-MARQUARDT (avec le complément de SCHUR) est parfaitement adaptée pour les problèmes avec quelques centaines voir milliers de paramètres, mais celle-ci s'accomode un peu moins bien au delà. Ce pourquoi, d'autres travaux proposent d'utiliser soit une optimisation par *Dog-Leg*, soit par les gradient conjugués.

Optimisations par *Dog-Leg*. LOURAKIS *et al.* [Lourakis 2005] ont suggéré qu'il était préférable de recourir à la méthode de *Dog-Leg* plutôt que de faire appel à un algorithme de LEVENBERG-MARQUARDT, d'une part parce que celle-ci semble montrer une accélération significative de la convergence (2.0 à 7.5 fois d'après l'article), et d'autre part, car l'implémentation des régions de confiances par *Dog-Leg* est plus adaptée pour contraindre l'ajustement (optimisation sous contraintes). De leur coté, CHEN *et al.* [Chen 2009] se servent d'une méthode de région de confiance plus avancée, le *Double Dog-Leg* [Dennis 1996], dans le cadre d'un problème de reconstruction de la trajectoire d'une caméra. L'emploi du *Double Dog-Leg* semble diminuer les

erreurs de localisation en comparaisons avec la technique de LEVENBERG-MARQUARDT. Les auteurs n'informent cependant pas des temps de convergences.

Optimisations par les gradients conjugués. Récemment, les méthodes des gradients conjugués avec préconditionnement [Byröd 2009, Jeong 2010, Agarwal 2010, Dellaert 2010] ont montré leur efficacité sur le problème d'ajustement de faisceaux de grande dimensions. Le principe de ces méthodes est d'appliquer en premier lieu un préconditionneur (de type JACOBI, GAUSS-SEIDEL [Byröd 2009], ou SSOR [1] [Jeong 2010, Agarwal 2010]) sur la matrice hessienne et de résoudre le problème par les gradients conjugués. Les auteurs de [Byröd 2009] proposent une approche multi-échelle dans laquelle ils appliquent un préconditionnement différent pour chaque groupe de paramètres.

Optimisations par recherche linéaire. Les algorithmes présentés (GAUSS-NEWTON, LEVENBERG-MARQUARDT, *Dog-Leg* ou les gradients conjugués) peuvent être significativement améliorés en leur ajoutant une technique de recherche linéaire [Nocedal 1999, Triggs 2000]. Néanmoins les techniques de recherche linéaire de l'état de l'art (voir la section 2.2.2.3) ont besoin d'une estimation assez précise de la matrice hessienne pour être performantes. Dans la communauté de vision par ordinateur, ces techniques sont très peu employées, notamment parce qu'elles sont itératives et assez coûteuses en temps de calcul. Il est ainsi souvent plus rapide de faire plus d'itérations sans recherche linéaire que moins d'itérations avec les méthodes de recherche linéaire de l'état de l'art.

4.1.2.2 Méthodes appropriées pour l'ajustement de faisceaux

L'importance grandissante de l'ajustement de faisceaux dans le domaine de la localisation et de la reconstruction par la vision a orienté les travaux de recherche vers la réduction de la complexité de ce problème (en $\mathcal{O}({N_p}^3)$). Différentes approches ont été proposées dans la littérature. Certaines exploitent la structure en réseau du problème pour réduire la complexité de l'ajustement de faisceaux, d'autres réduisent le nombre de paramètres à optimiser.

Méthodes exploitant la structure du problème. La complexité de résolution de l'ajustement de faisceaux est non seulement dépendante du nombre de paramètres à optimiser, mais elle est aussi affectée par la structure de la matrice hessienne de l'erreur de reprojection. Cette matrice est souvent très creuse et les méthodes de décomposition par bloc peuvent y être appliquées. Cela permet un gain considérable sur le temps de traitement puisque celles-ci (factorisation) ne sont effectuées que sur les bloc non nuls. Les méthodes de partitionnement agissent sur l'ordre des variables dans le but d'obtenir une structure (motif) plus adaptée, qui bénéficie d'un meilleur conditionnement [Triggs 2000].

Une des méthodes les plus couramment employée est le complément de SCHUR (section 3.2.3.3) décrit pour la première fois par GRANSHAW dans [Granshaw 1980]. Le principe est

1. Symmetric Successive Over Relaxation

de considérer que chaque point 3D reconstruit est indépendant des autres et ne dépend que des caméras qui l'observent. Cette structure particulière diminue fortement la complexité de l'ajustement de faisceaux (en $\mathcal{O}((N_p^c)^3)$), provenant principalement de l'inversion de la matrice hessienne approximée (seulement formée dans ce cas des blocs contenant les paramètres des caméras). Cette technique de réduction de complexité est aujourd'hui employée dans la plupart des applications utilisant l'ajustement de faisceaux [Triggs 2000, Mouragnon 2006, Konolige 2010] et notamment dans la bibliothèque libre développée par LOURAKIS *et al.* SBA (*Sparse Bundle Adjustment*) [Lourakis 2009].

Les auteurs de [Steedly 2003] se sont intéressé aux méthodes de partitionnement, qui tentent de réduire le problème d'ajustement de faisceaux en sous-problèmes plus petits et mieux conditionnés. Ils ont établit ainsi une technique de partitionnement spectrale, en regroupant les vecteurs propres de la matrice hessienne de l'erreur de reprojection correspondant aux petites valeurs propres. D'après les auteurs, la méthode s'avère être plus générale que les heuristiques sur la structure (*bottom-up*) et intègrent plus d'informations que les méthodes de partitionnement de matrice génériques.

Plus récemment, les auteurs de [Jeong 2010] ont proposé un ajustement de faisceaux très rapide, qui bénéficie d'une traitement optimisé par bloc, avec la bibliothèque de manipulation de matrices BLAS3. Ils proposent d'une part de réaliser une optimisation par Gradients Conjugués avec une factorisation LDL de la matrice hessienne des paramètres des caméras (complément de SCHUR) préconditionnée par bloc. Les paramètres doivent être ordonnés afin de conserver un motif par bloc adapté et les variables sont ensuite réordonnées avec l'heuristique MD (*Minimum Ordering*). Ils proposent de plus une heuristique nommée EPI (*Embedded Point Iterations*) dans laquelle les points sont optimisés à chaque nouvel incrément des paramètres des caméras. Cette heuristique est d'après les auteurs adaptée aux problèmes de grandes dimensions. L'ensemble de ces procédés améliore nettement la rapidité de l'ajustement de faisceaux.

Méthodes réduisant le nombre de paramètres à optimiser. D'autres travaux de recherche se sont focalisé sur la réduction du nombre de paramètres à optimiser simultanément, en ne raffinant par exemple qu'un sous ensemble des paramètres. La complexité en est ainsi réduite. Les méthodes diffèrent suivant le type de *Structure-from-Motion* considéré : *batch*, hiérarchique ou incrémental.

STEEDLY *et al.* [Steedly 2001] ont proposé une technique afin de réduire le nombre de paramètres à optimiser dans un problème de *Structure-from-Motion* incrémental dans lequel une nouvelle information (novatrice) est incorporée à une reconstruction pré-existante. Le principe est de rechercher les variables devant être mises à jour en détectant les relations entre les points et les caméras, et de lancer un ajustement de faisceaux seulement sur ces paramètres. Cette méthode, récursive et optimale d'ajustement de faisceaux incrémental est plus rapide car il ne nécessite pas de réaliser un ajustement de faisceaux global à chaque ajout d'une image. Cette technique peut cependant être assez gourmande en temps de calcul (récursivité) mais semble bien adaptée pour les ajustements larges.

L'approche publiée par SHUM *et al.* dans [Shum 1999] introduit la notion de image virtuelle

(*virtual frame*). Cette publication présente un algorithme de *Structure-from-Motion* hiérarchique dans lequel la reconstruction est effectuée par morceaux (segments) et un ajustement de faisceaux hiérarchique est exécuté afin d'optimiser les images virtuelles, représentes virtuelles de chaque segment. Le processus est itéré jusqu'à la reconstruction complète de la scène. Ainsi, le nombre de paramètres optimisés simultanément n'est jamais très important et dépend de la taille et du nombre de segments.

Une approche similaire est l'approche par squelette (*skeletal graph*) de SNAVELY *et al.* [Snavely 2008b]. Dans cette technique, ils réalisent d'une part un graphe reliant les caméras ayant un champ de vu commun, et réduisent ensuite celui-ci en conservant le squelette principale, représentant l'ensemble du graphe et conservant sa topologie. L'ensemble de l'information n'est donc pas complètement conservée. Seule l'information importante (redondantes) et de qualité (faible erreur de reprojection) est utilisée pour reconstruire une scène. Cette technique est particulièrement bien adaptée aux problèmes dont les données sont massives (milliers d'images).

Dans l'article [Ni 2007], une approche directe de *Structure-from-Motion* sur de larges données utilisant l'ensemble de celles-ci a été présentée. Le problème est subdivisé en sous-cartes optimisées séparément, puis fusionnées en une seulement et unique reconstruction. Les résultats montrent que la technique réduit le temps moyen de calcul nécessaire à estimation optimale de la reconstruction globale, principalement lorsque l'on construit de 2 à 8 sous-cartes (5, typiquement). Au delà, le temps de fusion des sous-cartes s'avère être très coûteux en temps de calcul.

Ces différentes techniques de réduction du nombre de variables sont principalement dédiées aux problèmes hors-lignes, c'est-à-dire que les ajustements peuvent prendre quelques minutes, heures, jours, voire plus. Dans le contexte d'un SLAM, il est préférable que l'optimisation ne durent que quelques centaines de millisecondes au maximum. Les investigations de MOURAGNON *et al.* et ENGELS *et al.* [Mouragnon 2006, Engels 2006] ont mené à l'établissement d'un ajustement de faisceaux local. Ces derniers considèrent dans leur ajustement de faisceaux seulement les caméras les plus récentes et leurs points 3D associés (vus). Cela permet de réduire considérablement les temps d'optimisation et peuvent ainsi l'employer dans une approche de SLAM à contrainte temps-réel.

Bien que ces méthodes apportent une certaine accélération à l'ajustement de faisceaux, elles utilisent toutes en interne un algorithme d'optimisation de type LEVENBERG-MARQUARDT et sont d'une complexité cubique.

4.1.3 Positionnement

Nos propositions présentées dans ce chapitre s'inscrivent dans le premier groupe des méthodes améliorant l'optimisation. Nous cherchons à améliorer l'efficacité des techniques d'optimisation en vision par ordinateur de manière générale, sans exploiter la structure du problème, ni la sélection des paramètres à optimiser. En revanche, ces propositions peuvent être entreprises en parallèle des méthodes du deuxième groupe, et bénéficier ainsi de leur réduction de complexité.

Comme nous venons de le voir, la majorité des techniques de l'état de l'art emploient en interne le principe de GAUSS-NEWTON, ou de sa variante (LEVENBERG-MARQUARDT) dans le but d'estimer un déplacement dans l'espace local des paramètres. Mais peu d'entre-elles

recherchent la meilleure amplitude pour ce déplacement à l'aide d'une méthode de recherche linéaire, qui en théorie améliorerait la qualité de la convergence. Sous leur forme générale, c'est-à-dire la recherche linéaire inexacte, ces méthodes sont extrêmement longues et n'apportent pas forcément un gain de temps. En effet, même si ces méthodes diminuent le nombre d'itérations nécessaires à l'algorithme d'optimisation, la recherche incrémentale de l'amplitude implique de nombreux calculs, avec notamment de nombreux appels à la fonction de coût.

Or, il est reconnu (section 2.2.2.3, [Nocedal 1999]) que les techniques de recherche linéaire améliorent la convergence des algorithmes d'optimisation non-linéaire de type descente de gradient, GAUSS-NEWTON ou LEVENBERG-MARQUARDT. Ajoutons de plus que pour ces derniers, c'est-à-dire pour les algorithmes d'optimisation de type GAUSS-NEWTON, la recherche de l'amplitude optimale est particulièrement utile lorsque l'approximation quadratique locale de la matrice hessienne de la fonction de coût est peu fidèle à la réalité, et lorsque la solution initiale du problème se trouve être éloignée de l'optimum. En effet, rappelons que la méthode de GAUSS-NEWTON lorsqu'on est près de la solution estime intrinsèquement la meilleure amplitude de déplacement (section 2.2.2.2).

C'est à partir de ces constats que nous avons choisi d'orienter les travaux de recherche sur la recherche linéaire. Nous proposons dans ce chapitre une méthode de recherche linéaire exacte, non itérative, adaptée aux problèmes de vision par ordinateur. Nous présentons dans un premier temps la méthode générale : la recherche linéaire algébrique et ses deux variantes globales et à deux dimensions, puis nous détaillons son utilisation dans le cadre d'un ajustement de faisceaux.

4.2 Recherche linéaire algébrique générique

Nous présentons dans cette section une nouvelle méthode de recherche linéaire non itérative, adaptée aux problèmes de reconstruction à partir du mouvement : la recherche linéaire algébrique (ALS : *Algebraic Line Search*). La méthode est déclinée en deux variantes : la recherche linéaire algébrique globale (*G-ALS*) et la recherche linéaire algébrique à deux dimensions (*Two way-ALS* ou juste *T-ALS*).

4.2.1 Principe général de la méthode

Dans le domaine de la vision par ordinateur, il est fréquent de minimiser un critère non-linéaire qui mesure une distance (2D, 3D, ...) entre des observations et leurs valeurs théoriques. Par exemple dans le problème d'estimation optimale de la matrice fondamentale, on minimise un terme d'erreur composé des distances entre les points 2D et les lignes épipolaires. Une autre exemple est l'ajustement de faisceaux dans lequel on s'intéresse à minimiser l'erreur de reprojection, formée des distances entre les positions 2D des points d'intérêt et leurs reprojections théoriques.

Habituellement, les erreurs de prédiction sont caractérisées par des distances euclidiennes, voire par des distances de Mahalanobis lorsque les incertitudes (variances) sur les mesures sont connues. L'intérêt de ces dernières est qu'elles bénéficient d'une justification probabiliste et

garantissent une estimation au maximum de vraisemblance ou au maximum *a posteriori* selon que l'on introduit ou non des informations *a priori* sur la densité de probabilité du modèle (voir la section 2.1).

Cependant, ces deux distances introduisent une non-linéarité supplémentaire au problème lorsque les entités sont exprimées en coordonnées affines (voir la section consacrée aux distances 1.4), c'est notamment le cas dans l'ajustement de faisceaux par exemple. Contrairement aux précédentes distances, la distance algébrique qui, rappelons-le, approxime la distance géométrique et est décrite dans la section 1.4.3, possède l'avantage de ne pas introduire de nouvelle non-linéarité au problème. C'est cet aspect particulier qui va nous intéresser.

4.2.1.1 Une méthode de recherche linéaire exacte

Nous proposons d'utiliser la distance algébrique dans une technique de recherche linéaire, et nous appelons cette technique la recherche linéaire algébrique.

Notre idée **n'est pas** d'utiliser la distance algébrique en tant que fonction de coût du problème (comme ce que propose HARTLEY [Hartley 1999] ou BARTOLI [Bartoli 2002]), nous gardons pour cela la distance géométrique. La distance algébrique est employée seulement dans le but de trouver une amplitude de déplacement efficace, optimale en distance algébrique, à chaque itération de la minimisation. La distance algébrique nous permet d'établir une approximation quadratique de la fonction objective $F(\mathbf{x})$, notée $F_{\mathcal{A}}(\mathbf{x})$ et définie par

$$F_{\mathcal{A}}(\mathbf{x}) = \sum_{z=[1,N_z]} d^2_{\mathcal{A}}(\hat{\mathbf{q}}_z, h^c_z(\mathbf{x})), \tag{4.1}$$

où $\hat{\mathbf{q}}_z$ est une observation (2D ou plus) et $h^c_z(\mathbf{x})$ est la fonction de prédiction de l'observation $\hat{\mathbf{q}}_z$ effectuée par la caméra, qui retourne le point théorique prévu par le modèle paramétré par $\mathbf{x}$.

Avec la définition de la distance algébrique (1.19) page 34, on observe que si la fonction de prédiction des observations est linéaire ou polynomiale (de degré n) par rapport aux paramètres du modèle $\mathbf{x}$, alors l'équation (4.1) se trouve être respectivement quadratique ou polynomiale (de degré $2n$). Ainsi, les problèmes d'optimisation minimisant ce terme d'erreur algébrique sont simplifiés. Nous verrons par la suite que c'est notamment le cas pour le problème de recherche linéaire algébrique, dont les solutions peuvent être calculées analytiquement dans certaines configurations, ou plus généralement estimées directement, en cherchant les racines du polynôme ainsi formé.

La méthode proposée est une méthode de recherche linéaire exacte, au sens de l'erreur algébrique (et inexacte au sens de l'erreur géométrique), et non itérative à la différence des principales méthodes de *line search* de l'état de l'art (section 2.2.2.3). Les amplitudes algébriques ainsi estimées sont considérées comme des approximations fiables (sous certaines conditions) des amplitudes optimales au sens de l'erreur géométrique ($\alpha^*_{\mathcal{E}} \approx \alpha^*_{\mathcal{A}}$).

La technique de recherche linéaire algébrique que nous proposons est constituée de plusieurs étapes successives, décrites dans la suite de ce chapitre :

– **Normalisation des données.** On normalise les données du problème par une normalisation isotropique (voir la section 4.2.2).

– **Calcul de l'erreur algébrique.** On calcule l'erreur algébrique du problème (et les divers coefficients du polynôme) étant donné la solution courante $\mathbf{x}^{(k)}$ et la direction de déplacement $\boldsymbol{\delta}^{(k)}$ (voir les sections 4.2.3 et 4.2.4).
– **Estimation de l'amplitude algébrique optimale du déplacement.** On résout le problème de recherche linéaire algébrique afin d'obtenir une ou plusieurs hypothèses d'amplitudes, optimales en distances algébriques (voir les sections 4.3.2 et 4.3.3 pour l'ajustement de faisceaux).
– **Critères de sélection et d'acceptation de l'amplitude.** On sélectionne la meilleure solution qui vérifie certains critères de qualité (voir les sections 4.2.3.2 et 4.2.4.2 pour l'ajustement de faisceaux).

4.2.1.2 Une méthode de recherche linéaire générique

La généricité de cette technique de recherche linéaire fait qu'elle peut facilement être intégrée dans n'importe quel algorithme de moindres carrés non-linéaire comme par exemple les algorithmes de descente de gradient ou de GAUSS-NEWTON, ou bien même dans les algorithmes de régions de confiances comme LEVENBERG-MARQUARDT (à l'image de ce que proposent NOCEDAL *et al.* [Nocedal 1992]).

De plus, la recherche linéaire algébrique peut être employée dans différents problèmes de vision par ordinateur : SLAM visuels, calcul de pose, triangulation ou tout autre problème minimisant une fonction objectif non-linéaire basée sur des distances géométriques.

Nous avons proposé deux approches différentes de la recherche linéaire algébrique : la recherche linéaire algébrique globale (*G-ALS*) et la recherche linéaire algébrique à deux dimensions (*Two way-ALS*). La recherche linéaire algébrique globale est une technique de recherche linéaire dont l'objectif est de rechercher une amplitude de déplacement efficace pour l'ensemble des paramètres. La variante *Two way-ALS* se propose d'estimer deux amplitudes différentes lorsqu'il est intéressant de séparer le vecteur de paramètres en deux parties. Nous décrivons plus précisément ces deux recherches linéaires algébriques, ainsi que leur application dans l'ajustement de faisceaux dans la suite de ce chapitre.

4.2.2 Normalisation des données

Comme nous venons de le mentionner, nous employons la distance algébrique afin de simplifier la fonction de coût du problème non-linéaire. Cette nouvelle fonction de coût algébrique est alors composée des erreurs algébriques entre les reprojections théoriques des points 2D et leurs observations. La distance algébrique, décrite dans la section 1.4.3, est une approximation de la distance euclidienne puisque :

$$d(\hat{\mathbf{q}}, \mathbf{q}) = \frac{d_{\mathcal{A}}(\hat{\mathbf{q}}, \mathbf{q})}{\hat{w}w}, \tag{4.2}$$

avec $\hat{\mathbf{q}}^\top = [\hat{x}\hat{y}\hat{w}]$ et $\mathbf{q}^\top = [xyw]$.

En conséquence, les solutions associées aux deux ajustements de faisceaux géométrique et algébrique ne sont pas nécessairement identiques. Pour limiter cette différence nous em-

ployons comme le préconisent les auteurs de [Hartley 1998, Chojnacki 2003] une normalisation isotropique des données. L'idée de la normalisation isotropique est de déplacer le centroïde des observations 2D de chaque image à l'origine du repère image, et de changer la moyenne des distances à l'origine afin que celle-ci devienne égale à $\sqrt{2}$. Nous modélisons les transformations isotropiques propres à chaque image i par les matrice $\mathcal{N}_i$, telles que

$$\mathcal{N}_i = \begin{pmatrix} 1 & 0 & -\bar{x}_i \\ 0 & 1 & -\bar{y}_i \\ 0 & 0 & \bar{s}_i \end{pmatrix}, \tag{4.3}$$

où $\bar{\tilde{\mathbf{q}}}_i = [\bar{x}_i \bar{y}_i]^\top$ est le centroïde des observations présentes sur l'image i et $\bar{s}_i$ est la moyenne des écarts des observations au centroïde. Les observations issues de la caméra sont donc normalisées par cette matrice : $\hat{\mathbf{q}}'_i = \mathcal{N}_i \hat{\mathbf{q}}_i$.

HARTLEY puis CHOJNACKI et BROOKS [Hartley 1998, Chojnacki 2003] montrent que l'erreur de reprojection algébrique donne des résultats satisfaisants, eu égard au fait que cette distance ne possède pas réellement de signification géométrique ou statistique. L'expérience montre cependant que sous cette normalisation adaptée – isotropique – des données du problème, les ajustements de faisceaux algébrique et géométrique donnent des solutions optimales à peu près similaires sur des problèmes de raffinement de la matrice essentielle notamment.

Cette normalisation des observations 2D n'est effectuée qu'une seule et unique fois, à l'initialisation de l'algorithme d'optimisation. Une fois que les données ont été ainsi normalisées, nous pouvons procéder à l'estimation de l'erreur de reprojection algébrique.

4.2.3 *G-ALS* : une recherche linéaire algébrique globale

4.2.3.1 Présentation

La recherche linéaire algébrique globale ou *G-ALS* (*Global Algebraic Line Search*) est la première version de recherche linéaire algébrique que nous proposons. Dans cette version, nous nous intéressons à l'estimation de l'amplitude de déplacement optimale au sens de l'erreur algébrique pour l'ensemble des paramètres composant le vecteur $\mathbf{x}$.

Le principe est d'utiliser la fonction de coût approximée (algébrique), qui va être ensuite employée dans un algorithme de recherche linéaire afin d'estimer l'amplitude optimale du déplacement. Le problème de recherche linéaire global avec cette nouvelle fonction de coût algébrique s'écrit alors :

$$\alpha_{\mathcal{A}}^{(k)^*} = \arg\min_{\alpha^{(k)}} F_{\mathcal{A}}(\mathbf{x}^{(k)} + \alpha^{(k)} \boldsymbol{\delta}^{(k)}), \tag{4.4}$$

où $F_{\mathcal{A}}$ est définie dans l'équation (4.1).

Une fois que l'algorithme de minimisation estime un déplacement $\boldsymbol{\delta}^{(k)}$ à partir du point courant $\mathbf{x}^{(k)}$ à l'itération k, notre objectif est de déterminer l'amplitude optimale $\alpha_{\mathcal{A}}^{(k)^*}$ le long de ce déplacement.

Le résolution du problème (4.4) dépend bien entendu du problème de vision par ordinateur auquel nous nous adressons (de $F^c(\mathbf{x})$), et de la paramétrisation du modèle ($\mathbf{x}$). Nous pouvons

simplement montrer que si la fonction de prédiction du problème $h^c(\mathbf{x})$ est polynomiale par rapport aux paramètres $\mathbf{x}$, alors ce problème de recherche linéaire algébrique global peut être résolu de manière non itérative et donc très rapidement. Nous avons étudié l'intégration de cette technique dans un ajustement de faisceaux et proposons la résolution du *G-ALS* dans ce cadre dans la section 4.3.2.

4.2.3.2 Critères de sélection et d'acceptation de l'amplitude

Dans la technique de recherche linéaire algébrique, nous employons deux types de critères sur l'amplitude du déplacement. Ces critères ainsi que l'ordre de vérification sont résumés dans l'algorithme 6.

Premièrement, l'algorithme *G-ALS* peut fournir plusieurs choix possibles d'amplitudes, et il faut alors choisir la meilleure, c'est l'objectif du critère pour les solutions multiples. De plus, nous proposons d'utiliser l'amplitude algébrique du déplacement comme une approximation de l'amplitude géométrique. Puisqu'il s'agit ici d'une approximation, nous ne faisons pas totalement confiance en cette valeur et nous proposons donc un critère dit d'amélioration afin de contrôler l'efficacité de l'amplitude algébrique.

Critère pour les solutions multiples. Comme nous venons de le voir, la méthode de recherche linéaire estime l'amplitude algébrique du déplacement en résolvant un polynôme de degré n. Or, les racines du polynôme peuvent être multiples (au maximum n) et il nous faut alors définir un critère mesurant la qualité d'une solution et ainsi choisir la meilleure des solutions.

Pour cela, nous réalisons un classement des solutions en fonction du gain qu'elles apportent sur l'erreur globale de reprojection géométrique. Par exemple si les racines du polynôme (4.12) sont au nombre de trois $\{\alpha^1_{\mathcal{A}}, \alpha^2_{\mathcal{A}}, \alpha^3_{\mathcal{A}}\}$ (comme dans le problème d'ajustement de faisceaux, où nous verrons que $n = 3$), nous estimons leurs impacts sur la fonction de coût géométrique avec la formule $F^c(\mathbf{x} + \alpha^i_{\mathcal{A}}\boldsymbol{\delta})$ et nous prenons ensuite l'amplitude minimisant cette erreur :

$$\alpha^*_{\mathcal{A}} = \arg \min_{\alpha=\{\alpha^1_{\mathcal{A}},\alpha^2_{\mathcal{A}},\alpha^3_{\mathcal{A}}\}} F^c(\mathbf{x} + \alpha\boldsymbol{\delta}). \tag{4.5}$$

L'idée ici est d'utiliser la fonction de coût classique et non approximée pour sélectionner la meilleure solution au regard de la vraie fonction objective. Cependant, en pratique, le polynôme (4.12) possède souvent une seule solution réelle et ce critère, assez consommateur de temps de calcul puisqu'il nécessite un appel à la fonction de coût pour chaque solution, est en pratique peu appelé.

Critères d'amélioration. Il est important de noter que la (pseudo-)distance algébrique est une approximation de la distance euclidienne (voir l'équation (4.2)). En conséquence, les minima des fonctions objectif euclidienne et algébrique ne sont pas obligatoirement identiques. L'expérience montre cependant que sous une normalisation adaptée des données, ces minima sont toutefois proches, comme le montre la figure 4.1. Pour éviter de choisir une amplitude qui détériorerait l'optimisation, nous procédons à un test dit d'amélioration. Ce test consiste d'une part à vérifier

que le déplacement avec l'amplitude choisie diminue bien la fonction de coût, et d'autre part que les conditions de WOLFE (voir la section 2.2.2.3) sont bien respectées. Parmi les deux critères de WOLFE, nous n'utilisons que le premier critère car il garantie une décroissance suffisante de l'erreur de reprojection. Le second critère ne semble pas nécessaire dans notre contexte étant donné la faible quantité d'hypothèses à tester (3 au maximum dans le cas de l'ajustement de faisceaux). La vérification de la diminution de l'erreur s'effectue en comparant la solution proposée, à la solution dont l'amplitude est l'unité ($\alpha = 1$) :

$$\alpha^*_{\mathcal{A}} = \arg \min_{\alpha=\{\alpha_{\mathcal{A}},1\}} F^c(\mathbf{x} + \alpha\boldsymbol{\delta}). \tag{4.6}$$

Le choix de la valeur unité n'est pas aléatoire. Rappelons que les algorithmes d'optimisation du type GAUSS-NEWTON estiment intrinsèquement l'amplitude du déplacement. Celui-ci étant d'autant plus efficace lorsqu'il est près de l'optimum, c'est-à-dire lorsque l'approximation quadratique de la fonction objectif est suffisamment précise. Ce critère nous permet ainsi de conserver l'efficacité du GAUSS-NEWTON lorsque l'on est près de la solution.

Algorithme 6 : `StepLengthSelectionGALS` : Sélection de la meilleure amplitude validant le premier critère de WOLFE.

- Entrées : Un ensemble d'hypothèses d'amplitudes $A = \{\ldots, \alpha_i, \ldots\}$
- Sorties : La meilleure amplitude du déplacement $\hat{\alpha}^*$

`// Sélection des hypothèses plus efficaces que l'amplitude unité`
$A' = \{\alpha^i_{\mathcal{A}} \in A\} \mid F^c(\mathbf{x} + \alpha^i_{\mathcal{A}}\boldsymbol{\delta}) < F^c(\mathbf{x} + \boldsymbol{\delta})$
si $Card(A') = 0$ **alors**
  `// L'amplitude unité est la meilleure solution`
  Retourner $\hat{\alpha}^* = 1$
sinon si $Card(A') = 1$ **alors**
  `// Une seule amplitude algébrique est plus efficace`
  Retourner $\hat{\alpha}^* = \alpha^i_{\mathcal{A}} \in A'$
sinon si $Card(A') > 1$ **alors**
  `// Plusieurs amplitudes algébriques sont efficaces.`
  `// Suppression des hypothèses ne validant pas la condition de WOLFE`
  $A'' = \{\alpha_i \in A'\} \mid \varepsilon_{\omega_1}(\alpha) = F^c(\mathbf{x}) + \omega_1\alpha\boldsymbol{\nabla}^\top\boldsymbol{\delta} - F^c(\mathbf{x} + \alpha\boldsymbol{\delta}) > 0$ `// Selection de l'amplitude maximisant la décroissance`
  Retourner $\hat{\alpha}^* = \arg\max_{A''} \varepsilon_{\omega_1}(\alpha)$

Tous ces critères sont appliqués séquentiellement sur l'hypothèse de l'amplitude algébrique estimée par la méthode *ALS* et permettent de choisir l'amplitude la plus efficace pour l'itération courante.

4.2.4 *Two way-ALS* : une recherche linéaire algébrique bi-dimensionnelle

Nous présentons dans cette section une variante de la méthode *G-ALS*, dans laquelle nous nous intéressons à estimer deux amplitudes distinctes pour deux parties du vecteur de déplacements.

4.2.4.1 Présentation

Dans certaines applications, les paramètres à optimiser peuvent être de différentes natures (mètres, radians, etc.) et peuvent de plus ne pas partager la même confiance, la même certitude. Par exemple dans l'ajustement de faisceaux, la structure de la scène et la trajectoire (position et orientation) de la caméra sont raffinées simultanément. Il semble alors intéressant de séparer la recherche de l'amplitude optimale du déplacement pour ces deux ensembles distincts.

Notre idée ici est de séparer la recherche linéaire en deux recherches linéaires, et cherchons simultanément les amplitudes de déplacement optimales pour les deux ensembles distincts de paramètres. Pour cela, nous allons employer une nouvelle fois l'approximation algébrique de la distance géométrique, à l'image de la méthode *G-ALS*. Nous appelons cette variante, la recherche linéaire algébrique à deux dimensions ou *Two way-ALS* (*Two-way Algebraic Line Search*).

Considérons que le vecteur de paramètres du modèle $\mathbf{x}^{(k)}$ se compose de deux sous-ensembles $\mathbf{P}$ et $\mathbf{Q}$ (par exemple les paramètres de la caméra et les paramètres de la scène) et que le vecteur de déplacement peut être décomposé en $\boldsymbol{\delta}^{(k)} = \{\boldsymbol{\delta}_{\mathbf{P}}^{(k)}, \boldsymbol{\delta}_{\mathbf{Q}}^{(k)}\}$ où $\boldsymbol{\delta}_{\mathbf{P}}^{(k)}$ et $\boldsymbol{\delta}_{\mathbf{Q}}^{(k)}$ sont les deux vecteurs de déplacement respectif. La recherche linéaire algébrique à deux dimensions consiste alors à résoudre le problème suivant :

$$\{\alpha_{\mathbf{P}}^{(k)^*}, \alpha_{\mathbf{Q}}^{(k)^*}\} = \arg \min_{\{\alpha_{\mathbf{P}}^{(k)}, \alpha_{\mathbf{Q}}^{(k)}\}} F_{\mathcal{A}}^{c}(\mathbf{x}^{(k)} + \begin{bmatrix} \alpha_{\mathbf{P}}^{(k)} \boldsymbol{\delta}_{\mathbf{P}}^{(k)} \\ \alpha_{\mathbf{Q}}^{(k)} \boldsymbol{\delta}_{\mathbf{Q}}^{(k)} \end{bmatrix}), \tag{4.7}$$

où $\alpha_{\mathbf{P}}^{(k)}$ et $\alpha_{\mathbf{Q}}^{(k)}$ sont les deux amplitudes recherchées pour les deux déplacements $\boldsymbol{\delta}_{\mathbf{P}}^{(k)}$ et $\boldsymbol{\delta}_{\mathbf{Q}}^{(k)}$.

Le résolution du problème (4.7) dépend de la fonction de coût $F^c(\mathbf{x})$ à minimiser et donc du problème de vision par ordinateur auquel nous nous adressons, et de la paramétrisation du modèle ($\mathbf{x}$). Nous pouvons toutefois montrer que si la fonction de prédiction du problème $h^c(\mathbf{x})$ de l'équation (4.1) est polynomiale par rapport aux paramètres $\mathbf{x}$, alors le problème de recherche linéaire algébrique à deux dimensions peut être résolu de manière non itérative. Nous avons étudié l'intégration de cette technique dans un ajustement de faisceaux et proposons la résolution du *Two way-ALS* dans ce cadre dans la section 4.3.2, en séparant les paramètres de la caméra des paramètres de la scène.

4.2.4.2 Critères de sélection et d'acceptation des amplitudes

À l'image de la méthode *G-ALS*, nous mettons en place une méthode pour valider les amplitudes proposées par la méthode. De plus, le *Two way-ALS* peut proposer plusieurs solutions (couples $\{\alpha_{\mathbf{P}}^{(k)}, \alpha_{\mathbf{Q}}^{(k)}\}$) possibles et il faut donc sélectionner la meilleure solution, le meilleur couple. Pour cela nous évaluons chacun des couples au regard de la fonction de coût géométrique

et sélectionnons alors le couple minimisant le plus celle-ci. La solution devra de plus respecter la première condition de WOLFE. Enfin, nous comparons cette proposition avec le couple unité $\{1, 1\}$ afin de conserver l'efficacité de la méthode de GAUSS-NEWTON lorsque l'on est près du minimum de la fonction de coût. La sélection du meilleur couple d'amplitudes est résumée dans l'algorithme 7.

Algorithme 7 : `StepLengthSelectionTALS` : Sélection de la meilleure amplitude validant le premier critère de WOLFE pour le *Two way-ALS*.

- Entrées : Un ensemble de couples hypothèses d'amplitudes $A = \{\ldots, (\alpha^i_{\mathbf{P}}, \alpha^i_{\mathbf{Q}}), \ldots\}$
- Sorties : Le meilleur couple amplitudes du déplacement $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*)$

```
// Sélection des hypothèses plus efficaces que l'amplitude
   unité
```

$A' = \{(\alpha^i_{\mathbf{P}}, \alpha^i_{\mathbf{Q}}) \in A\} \mid F^c(\mathbf{x} + \begin{bmatrix}\alpha^i_{\mathbf{P}}\boldsymbol{\delta}_{\mathbf{P}} \\ \alpha^i_{\mathbf{Q}}\boldsymbol{\delta}_{\mathbf{Q}}\end{bmatrix}) < F^c(\mathbf{x} + \begin{bmatrix}\boldsymbol{\delta}_{\mathbf{P}} \\ \boldsymbol{\delta}_{\mathbf{Q}}\end{bmatrix})$;

si $Card(A') = 0$ **alors**

```
// L'amplitude unité est la meilleure solution
```

Retourner $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*) = (1, 1)$

sinon si $Card(A') = 1$ **alors**

```
// Un seul couple d'amplitudes algébriques est retenu
```

Retourner $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*) = (\alpha^i_{\mathbf{P}}, \alpha^i_{\mathbf{Q}}) \in A'$

sinon si $Card(A') > 1$ **alors**

```
// Plusieurs couples sont possibles.
// Suppression des hypothèses ne validant pas la
   condition de WOLFE
```

$A'' = \{(\alpha^i_{\mathbf{P}}, \alpha^i_{\mathbf{Q}}) \in A' \mid \varepsilon_{\omega_1}(\alpha_{\mathbf{P}}, \alpha_{\mathbf{Q}}) =$
$F^c(\mathbf{x}) + \omega_1 \boldsymbol{\nabla}^\top \begin{bmatrix}\alpha_{\mathbf{P}}\boldsymbol{\delta}_{\mathbf{P}} \\ \alpha_{\mathbf{Q}}\boldsymbol{\delta}_{\mathbf{Q}}\end{bmatrix} - F^c(\mathbf{x} + \begin{bmatrix}\alpha_{\mathbf{P}}\boldsymbol{\delta}_{\mathbf{P}} \\ \alpha_{\mathbf{Q}}\boldsymbol{\delta}_{\mathbf{Q}}\end{bmatrix}) > 0\}$;

```
// Sélection du couple d'amplitudes maximisant la
   décroissance
```

Retourner $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*) = \arg\max_{A''} \varepsilon_{\omega_1}(\alpha_{\mathbf{P}}, \alpha_{\mathbf{Q}})$;

4.3 Recherche linéaire algébrique pour l'ajustement de faisceaux

Dans cette section nous proposons de résoudre les problèmes de recherche linéaire algébrique global et à deux dimensions, dans le cadre d'une optimisation non-linéaire très employée dans la problématique de reconstruction 3D : l'ajustement de faisceaux.

4.3.1 Principe général

La recherche linéaire algébrique est une méthode générique qui peut être employée dans divers problèmes de vision par ordinateur. L'ajustement de faisceaux (section 3.2) est un terrain propice qui peut facilement tirer partie de l'amélioration apportée par une recherche linéaire algébrique. En effet, l'ajustement de faisceaux est un problème complexe : un grand nombre de paramètres inter-dépendants sont optimisés simultanément. Chaque itération de l'algorithme d'optimisation est alors très importante en termes de temps de calcul. La méthode de recherche linéaire algébrique que nous proposons, a pour objectif de diminuer le nombre d'itérations et le temps nécessaire à la convergence de l'ajustement de faisceaux. la méthode se trouve être parfaitement bien adaptée à ce contexte.

Dans cette problématique, rappelons que la fonction objectif classique de l'ajustement de faisceaux est l'erreur de reprojection géométrique, définie par l'équation (3.25), page 79. Nous allons intégrer les deux variantes de recherche linéaire algébrique dans l'optimisation non-linéaire de l'ajustement de faisceaux.

Une fois que l'algorithme de minimisation estime à l'itération k un déplacement $\boldsymbol{\delta}^{(k)}$ à partir du point courant $\mathbf{x}^{(k)}$, notre objectif est de déterminer l'amplitude algébrique optimale $\alpha_{\mathcal{A}}^{(k)*}$ le long de ce déplacement. Pour cela, comme nous le présentons dans la section 4.2, nous employons l'erreur de reprojection algébrique définie par l'équation (3.28) page 79 pour estimer des amplitudes de déplacement approximées. Nous rappelons cette équation :

$$F_{\mathcal{A}}^{c}(\mathbf{x}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} \nu_j^i \varepsilon_j^i(\mathbf{x}) \quad \text{avec} \quad \varepsilon_j^i(\mathbf{x}) = d_{\mathcal{A}}(\hat{\mathbf{q}}_j^i, \mathbf{q}_j^i)^2. \tag{4.8}$$

Nous cherchons ensuite à résoudre un des deux problèmes de recherche linéaire algébrique globale ou à deux dimensions, définis respectivement par les équations (4.4) et (4.7).

À ce stade, nous devons différencier les ajustement de faisceaux calibrés et non calibrés. En effet, la paramétrisation locale μ_k définie dans la section 2.2.2 pour ces deux types d'ajustement de faisceaux va différer. Rappelons en effet que les techniques de recherche linéaire de l'état de l'art s'appliquent sur des fonctions de coût du type $F(\mathbf{x}^{(k)} + \alpha^{(k)}\boldsymbol{\delta}^{(k)})$ pour chaque incrément. Dans l'ajustement de faisceaux non calibré, la paramétrisation locale μ_k est réalisée par une simple addition de vecteurs. Mais dans l'ajustement de faisceaux calibré, les orientations de la caméra sont paramétrées par les matrices de rotation et les déplacements par les petits angles d'Euler. Nous devons alors réaliser une approximation de la paramétrisation locale μ_k pour revenir à un problème simplifié :

$$\mu_k(\mathbf{x}^{(k)}, \alpha^{(k)}\boldsymbol{\delta}^{(k)}) \approx \mathbf{x}^{(k)} + \alpha^{(k)}\boldsymbol{\delta}^{(k)}. \tag{4.9}$$

Cette approximation est présentée dans la section 4.3.4 sur la recherche linéaire algébrique pour l'ajustement de faisceaux calibré .

4.3.2 *G-ALS* pour l'ajustement de faisceaux non calibré

Dans l'ajustement non calibré, le vecteur de paramètres du modèle $\mathbf{x}^{(k)}$ est composé des matrices de projection $\mathcal{P}_i$ des caméras et de la structure de la scène : les points 3D $\mathbf{Q}_j$. Le vecteur de déplacement $\boldsymbol{\delta}^{(k)}$ est quant-à-lui constitué des déplacements de chaque matrice de projection $\boldsymbol{\delta}_{\mathbf{P}_i} = \text{vect}(\Delta_{\mathbf{P}_i})$ ainsi que des déplacements de chaque points 3D de la scène $\boldsymbol{\delta}_{\mathbf{Q}_j}$. La mise à jour du vecteur de paramètres du modèle $\mathbf{x}$ à l'itération suivante se réalise alors par une simple addition vectorielle : $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\boldsymbol{\delta}^{(k)}$.

La fonction de reprojection algébrique $F^c_{\mathcal{A}}$ de l'équation (4.8) dans le cadre d'un ajustement de faisceaux non calibré, et à l'itération k, s'écrit sous la forme suivante :

$$F^c_{\mathcal{A}}(\mathbf{x}^{(k)} + \alpha^{(k)}\boldsymbol{\delta}^{(k)}) = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left\| \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\mathcal{P}_i^{(k)} + \alpha^{(k)}\Delta_{\mathbf{P}_i}^{(k)})(\mathbf{Q}_j^{(k)} + \alpha^{(k)}\boldsymbol{\delta}_{\mathbf{Q}_j}^{(k)}) \right\|^2 . \tag{4.10}$$

Par la suite nous verrons que cette forme s'avère être très adaptée pour estimer l'amplitude du déplacement $\alpha^{(k)}$.

4.3.2.1 Résolution de la recherche linéaire algébrique

Lors de l'itération k de l'algorithme d'optimisation, nous disposons de l'estimation courante des paramètres du modèle ($\mathbf{x}^{(k)}$) et d'une direction de déplacement des paramètres $\boldsymbol{\delta}^{(k)}$ à réaliser. Nous nous intéressons ici à calculer l'amplitude du déplacement $\alpha^{(k)}$ en utilisant la fonction de reprojection algébrique. Si l'on développe l'équation (4.10), nous obtenons le résultat suivant (où l'on aura omis l'indice de l'itération k pour simplifier l'écriture) :

$$\begin{aligned}
F^c_{\mathcal{A}}(\mathbf{x} + \alpha\boldsymbol{\delta}) = &\sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \tfrac{1}{2} (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})\alpha^4 \\
&+ \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}\alpha^3 \\
+ \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \tfrac{1}{2} &\left(\left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j} + \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\right)^2 \right)\alpha^2 \\
&+ \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\alpha^1 \\
&+ \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)
\end{aligned} \tag{4.11}$$

Notons qu'il s'agit ici d'un polynôme de degré 4 en α. Les solutions du problème (4.4) correspondent aux amplitudes du déplacement annulant la dérivée de l'équation (4.11) par rapport à l'amplitude α :

$$\frac{\partial F^c_{\mathcal{A}}}{\partial \alpha}(\mathbf{x} + \alpha\boldsymbol{\delta}) = a_1\alpha^3 + b_1\alpha^2 + c_1\alpha + d_1 = 0, \tag{4.12}$$

avec

$$\begin{cases} a_1 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i 2(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ b_1 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i 3\left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j} \\ c_1 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left(\left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j} + \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j})\right)^2 \right) \\ d_1 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i \left(\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathsf{P}_i\mathbf{Q}_j\right)^\top \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\Delta_{\mathbf{P}_i}\mathbf{Q}_j + \mathsf{P}_i\boldsymbol{\delta}_{\mathbf{Q}_j}) \end{cases} . \quad (4.13)$$

Il s'agit d'un polynôme de degré 3 en α, dont les racines (au nombre maximum de trois) peuvent être facilement extraites de manière exacte (par la méthode de CARDAN ou SOTTA par exemple).

La figure 4.1 présente l'évolution des deux fonctions de reprojection géométrique et algébrique en fonction de l'amplitude α pour différentes itérations sur un problème réel. Nous pouvons observer dans les deux premières itérations que les minima des deux fonctions (symbolisés par un triangle rouge et un carré bleu), sont visuellement identiques, et correspondent pour la première itération à une amplitude de $\alpha^*_{\mathcal{A}} = 1.5$.

En approximant ainsi la fonction classique de reprojection par une distance algébrique (approximation quadratique) nous simplifions le problème non-linéaire en un problème quadratique dont les solutions peuvent être calculées analytiquement. Nous proposons ici une méthode de recherche rapide, non itérative contrairement à la majorité des techniques de l'état de l'art, et dont l'estimation s'effectue par la résolution du polynôme de degré 3 (4.12).

4.3.2.2 Résumé de l'algorithme *G-ALS* pour l'ajustement de faisceaux

Nous résumons les différentes étapes de la recherche linéaire algébrique globale dans l'algorithme `GlobalAlgebraicLineSearch` (8). Cet algorithme doit être intégré dans un algorithme de minimisation non-linéaire du type GAUSS-NEWTON (GAUSS-NEWTON, LEVENBERG-MARQUARDT, etc.), dans le cadre d'un ajustement de faisceaux. Nous proposons un exemple d'utilisation de cette méthode dans la section 4.4.

4.3.3 *Two way-ALS* pour l'ajustement de faisceaux non calibré

Dans l'ajustement de faisceaux, deux types de données sont optimisés : la trajectoire de la caméra (positions et orientations) et la structure de la scène (position des points 3D). Étant donné la nature différente de ces deux types de données du modèle, en unité voire en certitude, il semble intéressant de séparer la recherche de l'amplitude optimale du déplacement pour ces deux entités. Cela s'apparente alors à la technique d'alternance (section 3.2.4) où l'on optimise les caméras et la scène l'un après l'autre, mais dans notre cas, nous conservons les inter-dépendances entre les points et les caméras. Il a été montré [Hartley 2003a] que dans certaines applications (la factorisation notamment), l'alternance s'avère être plus efficace, plus rapide à converger que

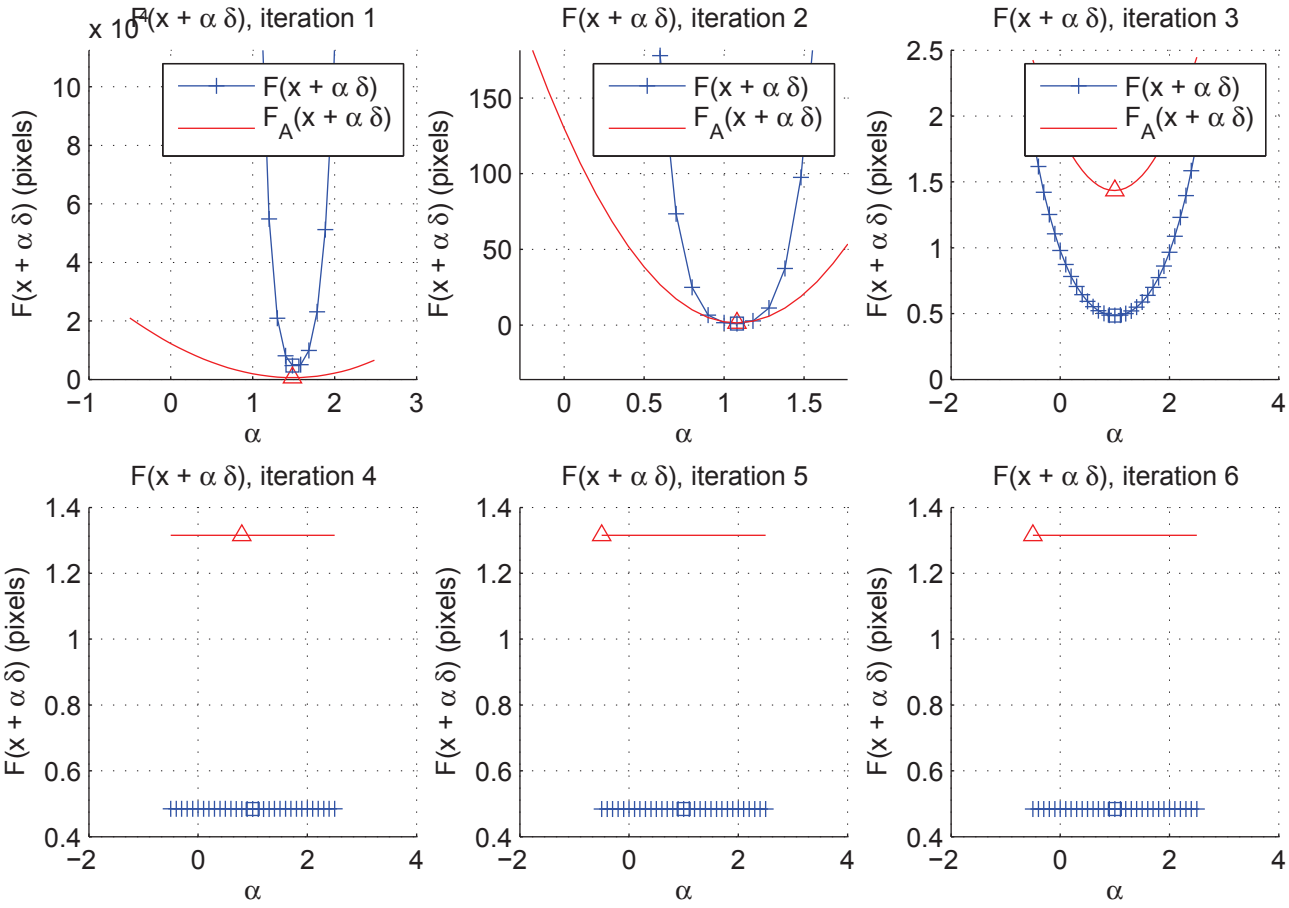


FIGURE 4.1 – Erreurs de reprojection $F(\alpha)$ (pixels) en fonction de l'amplitude α pour différentes itérations.

l'ajustement de faisceaux. De notre coté, au lieu de réaliser une alternance, nous proposons d'utiliser une variante de l'*ALS* pour l'ajustement de faisceaux : le *Two way-ALS* (décrit en section 4.2.4), afin de prendre en compte cette différence.

Le principe du *Two way-ALS* pour l'ajustement de faisceaux ici est de séparer la recherche linéaire en deux recherches linéaires. Nous cherchons dès lors simultanément les amplitudes de déplacement optimales pour les paramètres des caméras et pour les paramètres de la scène. Nous présentons dans cette section la résolution du *Two way-ALS* pour l'ajustement de faisceaux.

4.3.3.1 Résolution de la recherche linéaire algébrique

Après avoir normalisé les données (voir 4.2.2) , nous procédons au calcul de l'erreur algébrique étant donné un état $\mathbf{x}^{(k)}$ et un vecteur local de déplacement $\boldsymbol{\delta}^{(k)}$, composé de deux sous-vecteurs de déplacement pour les paramètres des caméras $\boldsymbol{\delta}_{\mathbf{P}}^{(k)}$ et les paramètres de la scène $\boldsymbol{\delta}_{\mathbf{Q}}^{(k)}$. La séparation des deux amplitudes pour les déplacements des caméras et de la scène donne

Algorithme 8 : `GlobalAlgebraicLineSearch` : Algorithme de recherche linéaire algébrique global *G-ALS*.

- Données : Les observations 2D normalisées $\hat{\mathbf{q}}_j^i = \mathcal{N}_i \hat{\mathbf{q}}_j^i$
- Entrées : Un point courant $\mathbf{x}^{(k)}$ et un déplacement $\boldsymbol{\delta}^{(k)}$
- Sorties : Une amplitude de déplacement optimale $\hat{\alpha_{\mathcal{A}}}^*$

`// Calcul des coefficients de l'ALS`
Calculer les paramètres $\{a_1, b_1, c_1, d_1\}$ avec les équations (4.13);
`// Résolution du polynôme`
Déterminer les solutions $A = \{\alpha_{\mathcal{A}}^1, \alpha_{\mathcal{A}}^2, \alpha_{\mathcal{A}}^3\}$ du polynôme de degré 3 (4.12).;
`// Vérifications des conditions et sélection de la meilleure amplitude`
Retourner $\hat{\alpha}_{\mathcal{A}}^*$ =`StepLengthSelectionGALS` (A);

l'équation de reprojection algébrique suivante :

$$F_{\mathcal{A}}^c(\mathbf{x}^{(k)} + \begin{bmatrix} \alpha_{\mathbf{P}}^{(k)} \boldsymbol{\delta}_{\mathbf{P}}^{(k)} \\ \alpha_{\mathbf{Q}}^{(k)} \boldsymbol{\delta}_{\mathbf{Q}}^{(k)} \end{bmatrix}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} \nu_j^i \left\| \mathcal{S}[\hat{\mathbf{q}}_j^i]_\times (\mathcal{P}_i^{(k)} + \alpha_{\mathbf{P}}^{(k)} \Delta_{\mathbf{P}_i}^{(k)})(\mathbf{Q}_j^{(k)} + \alpha_{\mathbf{Q}}^{(k)} \boldsymbol{\delta}_{\mathbf{Q}_j}^{(k)}) \right\|^2 . \quad (4.14)$$

Nous cherchons alors les deux amplitudes de déplacement optimales $\alpha_{\mathbf{P}}^{(k)}$ et $\alpha_{\mathbf{Q}}^{(k)}$, pour les déplacements respectivement des caméras $\boldsymbol{\delta}_{\mathbf{P}}^{(k)}$ et de la scène (points 3D) $\boldsymbol{\delta}_{\mathbf{Q}}^{(k)}$. Nous omettons l'indice de l'itération pour simplifier les notations et nous obtenons le problème suivant :

$$\{\alpha_{\mathbf{P}}^*, \alpha_{\mathbf{Q}}^*\} = \arg \min_{\{\alpha_{\mathbf{P}}, \alpha_{\mathbf{Q}}\}} F_{\mathcal{A}}^c(\mathbf{x} + \begin{bmatrix} \alpha_{\mathbf{P}} \boldsymbol{\delta}_{\mathbf{P}} \\ \alpha_{\mathbf{Q}} \boldsymbol{\delta}_{\mathbf{Q}} \end{bmatrix}). \quad (4.15)$$

Les dérivées partielles de la fonction de reprojection algébrique en $\alpha_{\mathbf{P}}$ et $\alpha_{\mathbf{Q}}$ nous amènent à résoudre le système suivant :

$$\frac{\partial F_{\mathcal{A}}^c}{\partial \alpha_{\mathbf{P}}}(\mathbf{x} + \begin{bmatrix} \alpha_{\mathbf{P}} \boldsymbol{\delta}_{\mathbf{P}} \\ \alpha_{\mathbf{Q}} \boldsymbol{\delta}_{\mathbf{Q}} \end{bmatrix}) = 0 \quad \text{et} \quad \frac{\partial F_{\mathcal{A}}^c}{\partial \alpha_{\mathbf{Q}}}(\mathbf{x} + \begin{bmatrix} \alpha_{\mathbf{P}} \boldsymbol{\delta}_{\mathbf{P}} \\ \alpha_{\mathbf{Q}} \boldsymbol{\delta}_{\mathbf{Q}} \end{bmatrix}) = 0 \ . \quad (4.16)$$

Lorsque l'on développe ce système, nous obtenons les équations suivantes :

$$\begin{array}{l} \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} 2a_2 + 2c_2\alpha_{\mathbf{P}} + 2d_2\alpha_{\mathbf{P}}^2 + 2\alpha_{\mathbf{Q}}\alpha_{\mathbf{P}}(2e_2 + f_2\alpha_{\mathbf{P}}) + 2g_2\alpha_{\mathbf{Q}} = 0 \\ \sum_{i=1}^{N_m} \sum_{j=1}^{N_n} 2b_2 + 2c_2\alpha_{\mathbf{Q}} + 2e_2\alpha_{\mathbf{Q}}^2 + 2\alpha_{\mathbf{Q}}\alpha_{\mathbf{P}}(2d_2 + f_2\alpha_{\mathbf{Q}}) + 2h_2\alpha_{\mathbf{P}} = 0 \end{array} , \quad (4.17)$$

avec les paramètres

$$\begin{cases} a_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{P}_i\mathbf{Q}_j)^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathcal{P}_i \boldsymbol{\delta}_{\mathbf{Q}_j}) \\ b_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{P}_i\mathbf{Q}_j)^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\mathbf{Q}_j) \\ c_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathcal{P}_i \boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\mathbf{Q}_j) + (\mathcal{P}_i\mathbf{Q}_j)^\top (\Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ d_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\mathbf{Q}_j)^\top (\Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ e_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathcal{P}_i \boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ f_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\Delta_{\mathbf{P}_i}\boldsymbol{\delta}_{\mathbf{Q}_j}) \\ g_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathcal{P}_i \boldsymbol{\delta}_{\mathbf{Q}_j})^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \mathcal{P}_i \boldsymbol{\delta}_{\mathbf{Q}_j}) \\ h_2 = \sum_{i=1}^{N_m}\sum_{j=1}^{N_n} \nu_j^i (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}Q)^\top (\mathcal{S}[\hat{\mathbf{q}}_j^i]_\times \Delta_{\mathbf{P}_i}\mathbf{Q}_j) \end{cases} . \tag{4.18}$$

La résolution symbolique de ce système de polynômes avec le logiciel *Maple*, qui utilise en interne les bases de GRÖBNER, nous donne les solutions optimales suivantes :

$$\{\alpha_{\mathbf{P}}^*, \alpha_{\mathbf{Q}}^*\} = \{p_2, \frac{-a_2 - c_2 p_2 - d_2 p_2^2}{g_2 + 2e_2 p_2 + f_2 p_2^2}\}, \tag{4.19}$$

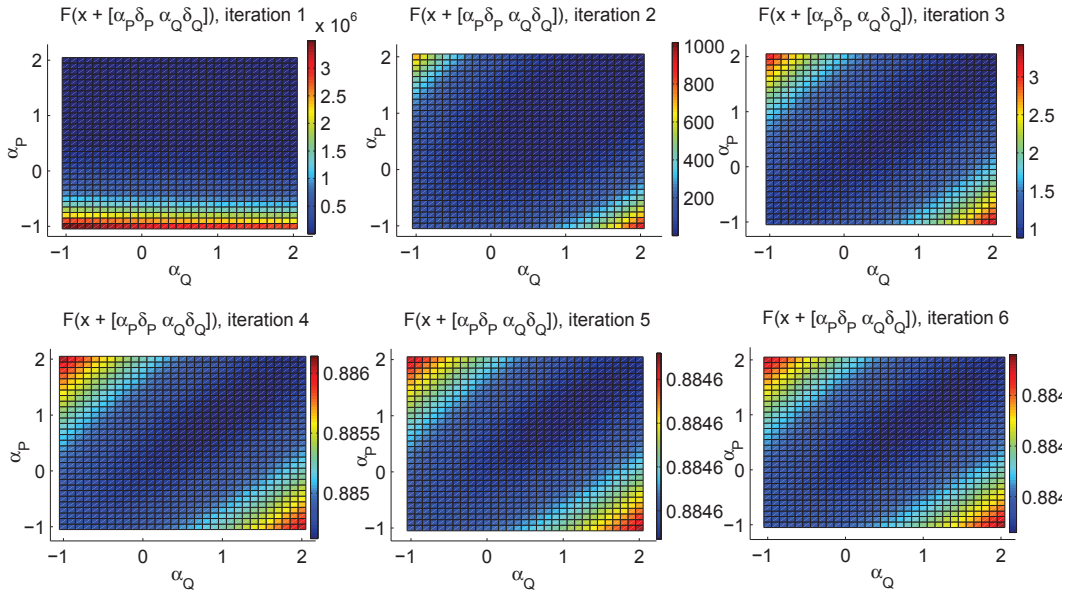
où $p_2 = racines(\Upsilon(X))$ sont les racines d'un polynôme de degré 5 : $\Upsilon : \mathbb{R} \to \mathbb{R}$, tel que

$$\begin{aligned} \Upsilon(X) = (-h_2 f_2^2 + d_2^2 f_2)X^5 \\ +(3e_2 d_2^2 - 4h_2 e_2 f_2 + c_2 d_2 f_2 - b_2 f_2^2)X^4 \\ +(-4h_2 e_2^2 - 4b_2 e_2 f_2 - 2h_2 f_2 g_2 + 4c_2 d_2 e_2 + 2d_2^2 g_2)X^3 \\ +(c_2^2 e_2 - 4h_2 e_2 g_2 - 4b_2 e_2^2 + 2e_2 a_2 d_2 - 2b_2 f_2 g_2 - c_2 a_2 f_2 + 3c_2 d_2 g_2)X^2 \\ +(c_2^2 g_2 + 2d_2 a_2 g_2 - 4b_2 e_2 g_2 - f_2 a_2^2 - h_2 g_2^2)X \\ -b_2 g_2^2 - e_2 a_2^2 + c_2 a_2 g_2 \end{aligned} .$$

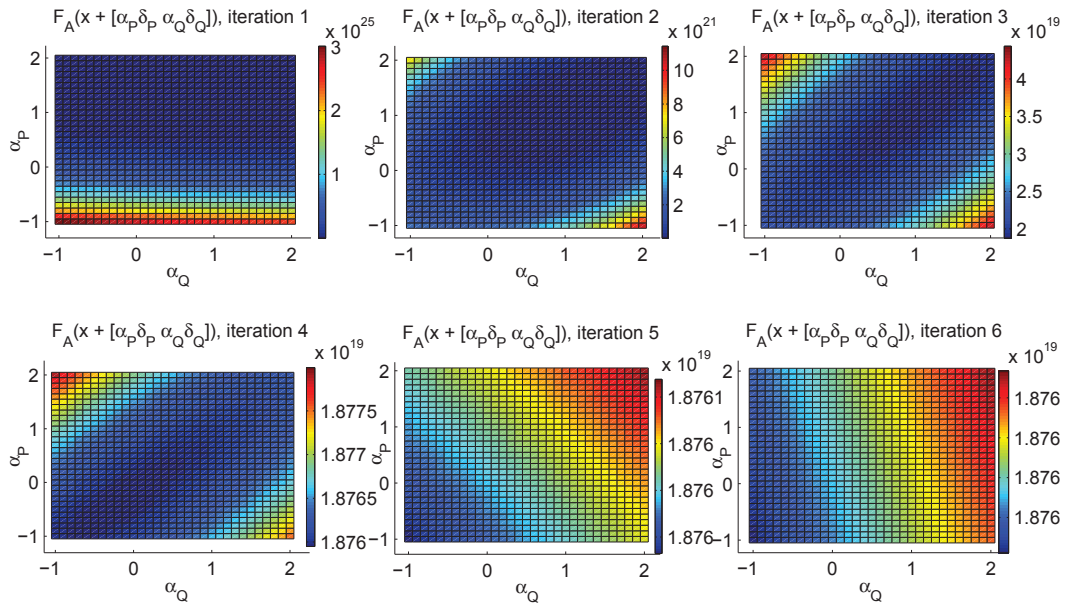
Le problème de la recherche linéaire à deux dimensions peut donc être résolu en déterminant les racines d'un polynôme de degré 5, correspondant à plusieurs hypothèses d'amplitudes du déplacement des paramètres des caméras. Les hypothèses d'amplitudes des déplacements de la scène correspondantes sont ensuite déduites des paramètres des caméras. En pratique, nous avons employé la fonction `roots` du logiciel *Matlab* afin de trouver ces solutions.

L'erreur de reprojection algébrique. La figure suivante 4.2 illustre les erreurs de reprojection géométrique et algébrique en fonction des deux paramètres d'amplitudes $\alpha_{\mathbf{P}}$ et $\alpha_{\mathbf{Q}}$ lors de plusieurs itérations consécutives d'un ajustement de faisceaux. Nous pouvons remarquer que les formes des fonctions sont assez similaires et partagent les mêmes zones minimales, d'autant plus

lors des premières itérations. Cela signifie que l'erreur de reprojection algébrique est une bonne approximation de l'erreur de reprojection géométrique et que l'on peut donc estimer les deux amplitudes optimales au sens de l'erreur géométrique à l'aide des amplitudes algébriques.



(a) Erreur de reprojection géométrique



(b) Erreur de reprojection algébrique

FIGURE 4.2 – Erreur de reprojection géométrique (a) et algébrique (b) pour différentes valeurs d'amplitudes de déplacements (caméra $\alpha_{\mathbf{P}}$ et scène $\alpha_{\mathbf{Q}}$), pour 6 itérations.

Nous pouvons aussi remarquer dans la figure 4.2(a), lors de la première itération de l'ajuste-

ment de faisceaux, que les deux amplitudes $\alpha_{\mathbf{P}}$ et $\alpha_{\mathbf{Q}}$ ne partagent pas le même comportement. En effet dans le cas contraire, nous aurions une forme d'ellipse dont l'axe principal serait proche de $\alpha_{\mathbf{P}} = \alpha_{\mathbf{Q}}$, ce que l'on retrouve à partir de la seconde itération. Cela montre qu'il est bénéfique de séparer la recherche de l'amplitude pour les deux déplacements, ce que propose cette variante.

Enfin, on peut voir que les deux amplitudes ne possèdent pas nécessairement leur minimum au même endroit, et c'est notamment le cas sur la dernière itération où l'erreur algébrique semble posséder un minimum négatif, dans la direction opposée à celle proposée par l'algorithme de minimisation.

4.3.3.2 Résumé de l'algorithme *T-ALS* pour l'ajustement de faisceaux

Nous résumons les étapes de la méthode *Two way-ALS* dans l'algorithme `TwoWayAlgebraicLineSearch` (9). Cet algorithme doit être intégré dans un algorithme de minimisation non-linéaire du type GAUSS-NEWTON (GAUSS-NEWTON, LEVENBERG-MARQUARDT, etc.), dans le cadre d'un ajustement de faisceaux. Nous proposons un exemple d'utilisation de cette méthode dans la section suivante.

Algorithme 9 : `TwoWayAlgebraicLineSearch` : Algorithme de recherche linéaire algébrique *Two way-ALS*

- Données : Les observations 2D normalisées $\hat{\mathbf{q}}'_i = \mathcal{N}_i \hat{\mathbf{q}}_i$
- Entrées : Un point courant $\mathbf{x}$ et deux déplacements $\boldsymbol{\delta}_{\mathbf{P}}, \boldsymbol{\delta}_{\mathbf{Q}}$
- Sorties : Un couple d'amplitudes de déplacement optimal (en erreur algébrique) $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*)$

`// Calcul des coefficients de l'ALS`
Calculer les paramètres $\{a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2\}$ avec les équations (4.18);
`// Résolution du polynôme`
Déterminer les racines $A = \{(\alpha^1_{\mathbf{P}}, \alpha^1_{\mathbf{Q}}), \ldots, (\alpha^5_{\mathbf{P}}, \alpha^5_{\mathbf{Q}})\}$ du polynôme de degré 5 (4.3.3.1) dans l'intervalle $(\alpha^i_{\mathbf{P}}, \alpha^i_{\mathbf{Q}}) \in \mathbb{R}^2$`// Vérifications des conditions et sélection du meilleur couple`
Retourner $(\hat{\alpha_{\mathbf{P}}}^*, \hat{\alpha_{\mathbf{Q}}}^*) =$`StepLengthSelectionTALS` (A);

4.3.4 Ajustement de faisceaux calibré

La principale différence entre les ajustements de faisceaux calibrés et non calibrés réside dans le choix des paramètres des caméras à optimiser. Lorsque les caméras sont calibrées, c'est-à-dire que les paramètres intrinsèques sont connus, l'ajustement de faisceaux est dit calibré. Dans ce cas, les paramètres des caméras ne sont plus définis par leur matrice de projection $\mathcal{P}_i$, mais par leurs informations de pose (orientation et position). Le vecteur de déplacement $\boldsymbol{\delta}$ contient quant-à-lui pour chaque caméra c_i, un déplacement en petits angles d'Euler $\boldsymbol{\delta}_{\mathbf{e}_i}$ et une translation relative $\boldsymbol{\delta}_{\mathbf{t}_i}$, ainsi que les déplacements de la scène $\boldsymbol{\delta}_{\mathbf{Q}_j}$. Si l'on désire retrouver la forme de

l'équation (4.10) afin de pouvoir employer les mêmes algorithmes d'estimation d'amplitude $\alpha^{(k)}$ dans les ajustement de faisceaux calibrés que dans les ajustement de faisceaux non calibrés, nous devons estimer une approximation du déplacement de la matrice de projection $\Delta_{\mathbf{P}_i}^{(k)}$ afin d'avoir l'équation de mise à jour dans le cas non calibré telle que : $\mathcal{P}_i^{(k+1)} = \mathcal{P}_i^{(k)} + \alpha^{(k)}\Delta_{\mathbf{P}_i}^{(k)}$. Pour cela, nous approximons cette matrice $\Delta_{\mathbf{P}_i}^{(k)}$ par :

$$\Delta_{\mathbf{P}_i}^{(k)} \approx \mathcal{K}_i\mathcal{R}_i^{(k)} \begin{pmatrix} [\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}]_\times & \boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \end{pmatrix}. \tag{4.20}$$

En effet, la mise à jour de la matrice de projection à l'itération k selon la paramétrisation locale μ_k dans le cas calibré s'écrit :

$$\begin{aligned}
\mathcal{P}_i^{k+1} &= \mathcal{P}_i^k \begin{pmatrix} \text{rot}_\text{e}\left(\alpha^{(k)}\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}\right) & \alpha^{(k)}\boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \\ \mathbf{0} & 1 \end{pmatrix} \\
&= \mathcal{K}_i^{(k)} \begin{pmatrix} \mathcal{R}_i^{(k)} & \mathbf{t}_i^{(k)} \end{pmatrix} \begin{pmatrix} \text{rot}_\text{e}\left(\alpha^{(k)}\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}\right) & \alpha^{(k)}\boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \\ \mathbf{0} & 1 \end{pmatrix} \\
&= \mathcal{K}_i \begin{pmatrix} \mathcal{R}_i^{(k)}\,\text{rot}_\text{e}\left(\alpha^{(k)}\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}\right) & \mathbf{t}_i^{(k)} + \alpha^{(k)}\mathcal{R}_i^{(k)}\boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \end{pmatrix}.
\end{aligned}$$

Or, en utilisant l'approximation classique des petits angles $\text{rot}_\text{e}(\mathbf{v}) \approx \mathcal{I}_3 + [\mathbf{v}]_\times$, nous obtenons :

$$\begin{aligned}
\mathcal{P}_i^{(k+1)} &\approx \mathcal{K}_i \begin{pmatrix} \mathcal{R}_i^{(k)} + \alpha^{(k)}\mathcal{R}_i^{(k)}[\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}]_\times & \mathbf{t}_i^{(k)} + \alpha^{(k)}\mathcal{R}_i^{(k)}\boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \end{pmatrix} \\
&\approx \mathcal{P}_i^{(k)} + \alpha^{(k)}\mathcal{K}_i\mathcal{R}_i^{(k)} \begin{pmatrix} [\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}]_\times & \boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \end{pmatrix} \\
&\approx \mathcal{P}_i^{(k)} + \alpha^{(k)}\Delta_{\mathbf{P}_i}^{(k)},
\end{aligned}$$

avec $\Delta_{\mathbf{P}_i}^{(k)} \approx \mathcal{K}_i\mathcal{R}_i^{(k)} \begin{pmatrix} [\boldsymbol{\delta}_{\mathbf{e}_i}^{(k)}]_\times & \boldsymbol{\delta}_{\mathbf{t}_i}^{(k)} \end{pmatrix}$.

Cette approximation de $\Delta_{\mathbf{P}_i}^{(k)}$ nous permet de conserver l'équation (4.10) dans le cas des ajustements calibrés. Nous appliquons ensuite les même procédures de calcul de l'amplitude de déplacement que dans le cas d'un ajustement de faisceaux non calibré.

4.4 Résultats expérimentaux

4.4.1 Préliminaires

Nous présentons dans cette section les résultats expérimentaux des deux approches de recherche linéaire, les recherche linéaire algébrique globale et à deux dimensions sur des données synthétiques et réelles. Ces évaluations sont obtenues dans le cadre de deux types d'ajustements de faisceaux calibré et non calibré.

4.4.1.1 Contexte de l'évaluation

L'ensemble de ces expérimentations a été réalisé à l'aide du logiciel MATLAB 7.5.0 sur un ordinateur fixe doté d'une mémoire RAM de 3Go et d'un processeur Intel XEON à double cœurs, chacun cadencés à 3.00 GHz. Nous avons implémenté la majorité des algorithmes, en langage Matlab, à partir d'une première structure originellement conçue par NICOLAS GUILBERT et dont le code source est disponible sur Internet [2].

4.4.1.2 Génération de données de synthèse

Nous générons aléatoirement 500 points 3D dans un cube de 6 mètres, et 30 caméras sont positionnées circulairement autour à une distance de 20 mètres par rapport au centre du cube. Les caméras possèdent des paramètres intrinsèques identiques et connus : une focale de 1000 pixels, un centre de projection situé au centre de l'image et les pixels sont considérés carrés (facteur de biais nul). Les projections 2D de la scène dans les différentes images sont ensuite estimées, en éliminant les points situés en dehors des images (de dimensions 640×480 pixels). Ces projections 2D sont altérés par un bruit blanc Gaussien dont l'écart type est $\sigma = 1$ pixel. Nous retirons de plus les images n'ayant pas assez de projections (typiquement 10 points 2D) afin d'éviter les configurations instables pour l'optimisation de la pose.

La figure 4.3 illustre la configuration de la scène simulée et la figure 4.4 montre la localisation des 100 premiers points 2D dans les 6 premières images et les reprojections 3D de la scène après l'étape d'initialisation (sans optimisation non-linéaire).

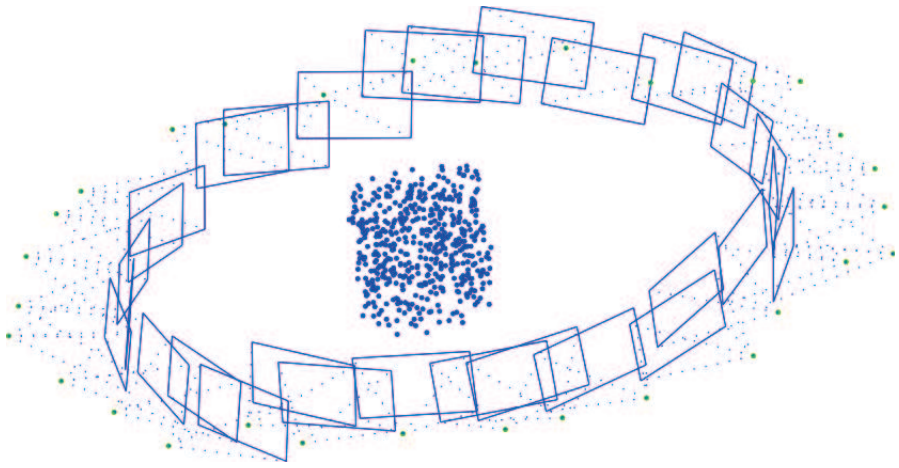


FIGURE 4.3 – Illustration de la scène synthétique. Les caméras sont positionnées autour de la scène et se dirigent vers le centre de celle-ci, le centre du repère.

2. http://www.maths.lth.se/matematiklth/personal/nicolas/octave-vision.html

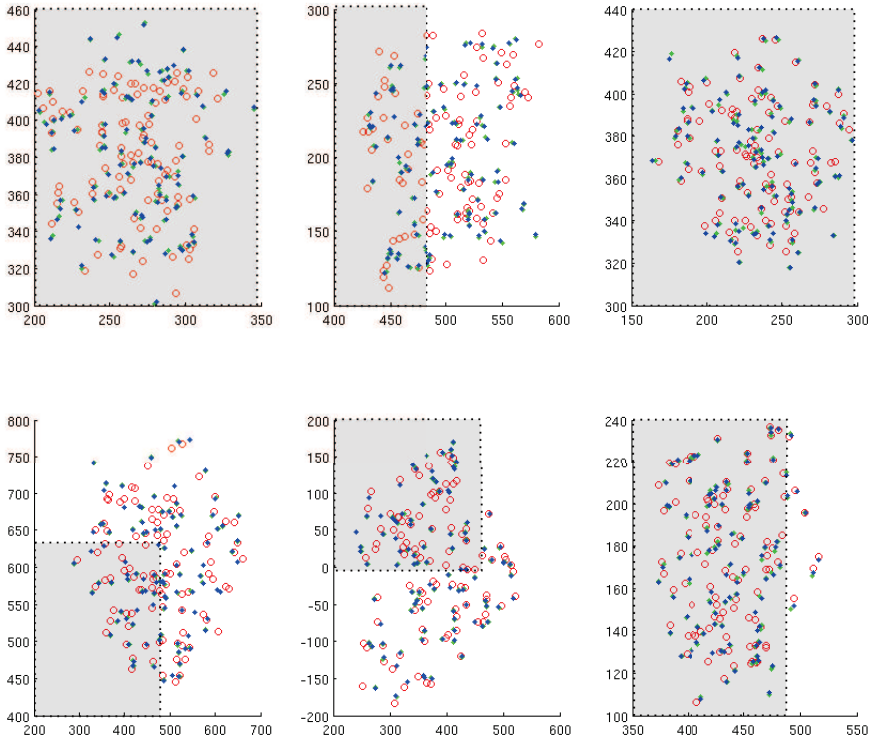


FIGURE 4.4 – Projections des points dans les images synthétiques (zones sombres). En vert il s'agit des points réels, en bleu les points bruités et en rouge les points 3D de la scène reprojetés après l'étape d'initialisation.

4.4.1.3 Estimation de la solution initiale

Données synthétiques. Dans le contexte de l'expérimentation sur des données synthétiques, le vecteur initial $\mathbf{x}^{(0)}$ constituant le modèle à raffiner est généré à partir des vraies positions des points 3D et des vraies poses des caméras, auxquelles nous avons ajouté des bruits blancs Gaussiens, d'écarts types $\sigma_Q = 0.5$ m (≈ 10 %) pour la scène et $\sigma_t = 1$ m ($\frac{1}{20}$ème du champ) et $\sigma_R = 15°$ pour respectivement la position et l'orientation des caméras.

Données réelles. Sur les données réelles, nous avons testé les approches dans deux applications qui sont le *batch Structure-from-Motion* 4 et le *Structure-from-Motion* incrémental 5.

***Batch Structure-from-Motion*.** Dans le *batch Structure-from-Motion*, la détection et l'appariement des points d'intérêts dans les images a été effectuée à l'aide du détecteur et descripteur SIFT, dont une implémentation est proposée par DAVID LOWE et disponible sur Internet [3]. Les

3. http://www.cs.ubc.ca/~lowe/keypoints/

appariements sont ensuite estimés de manière robuste en estimant les matrices fondamentales entre chaque images et en éliminant les appariements incohérents (*outliers*), dont la distance du point 2D à la ligne épipolaire est supérieure à un seuil de 0.8 pixel. L'estimation initiale de la scène et de la trajectoire de la caméra a été réalisée par la méthode proposée par TARDIF *et al.*, dont nous avons récupéré une implémentation partielle. Parmi les contraintes proposées par les auteurs de [Tardif 2007], nous avons choisi d'utiliser les contraintes de caméras (CCC : *Camera Closure Constraints*).

***Structure-from-Motion* incrémental.** Enfin, les dernières expériences ont été effectuées à partir de solutions initiales estimées par un SLAM monoculaire, écrit originellement en C++ par MOURAGNON *et al.* et amélioré par l'équipe LVIC du CEA LIST.

Les solutions initiales ainsi formée selon ces deux procédés sont ensuite envoyées aux différents algorithmes d'ajustement de faisceaux et forment ainsi la base commune pour réaliser une comparaison des propositions avec l'état de l'art.

4.4.1.4 Méthodes de l'état de l'état de l'art comparées

Nous avons comparé les algorithmes de recherche linéaire algébrique à différentes méthodes d'optimisation non-linéaire de l'état de l'art. La plupart de celles-ci ont été implémentées par nos soins. Chacun des algorithmes est nommé de la manière suivante :

1. `BA-LM` est une implémentation classique de l'ajustement de faisceaux utilisant LEVENBERG-MARQUARDT et les règles de mise à jour du paramètre de régularisation de [Hartley 2003b], décrit dans la section 2.2.2.4. La matrice de régularisation $\mathcal{D}$ est définie par $\mathcal{D} = \mathrm{diag}({\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta)$.
2. `BA-Nielsen` est la technique proposée par NIELSEN dans [Nielsen 1999]. Celle-ci utilise aussi LEVENBERG-MARQUARDT mais avec notamment une règle distincte de mise à jour du paramètre de régularisation. L'implémentation est inspirée de celle fournit par NIELSEN [4]. La matrice de régularisation $\mathcal{D}$ est définie par $\mathcal{D} = \mathcal{I}$. Les paramètres suivant ont été utilisés : $\nu = 2$, pas de décomposition de CHOLESKY,
3. `BA-Powell` est un ajustement de faisceaux basé sur la méthode de *Dog-Leg* [Powell 1970], présenté dans la section 2.2.2.4. Nous avons implémenté cet algorithme en nous inspirant des travaux de LOURAKIS *et al.* [Lourakis 2005].
4. `BA-LM+LS-FJNT` est l'algorithme de recherche linéaire exacte proposé par FRANDSEN *et al.* [Frandsen 1999] décrit dans la section 2.2.2.3. Nous avons pris l'implémentation de NIELSEN, téléchargable sur Internet [5]. Cette méthode de recherche linéaire est l'une des plus précises dans l'estimation de l'amplitude optimale locale des déplacements, mais au détriment d'une consommation excessive du temps CPU. Elle nous servira donc de repère

4. http://www2.imm.dtu.dk/~hbn/Software/damp_newton.m
5. http://www2.imm.dtu.dk/~hbn/immoptibox

en nous indiquant ce qu'une méthode précise mais lente pourrait apporter à la minimisation. Nous employons de plus parfois la version *soft* de la méthode qui est une implémentation de la recherche linéaire par ajustement (voir section 2.2.2.3 page 45).

5. `G-ALS` et `T-ALS` sont les heuristiques basées sur la recherche linéaire algébrique, décrites aux sections 4.3.2 et 4.3.3.

4.4.2 Résultats sur des données synthétiques

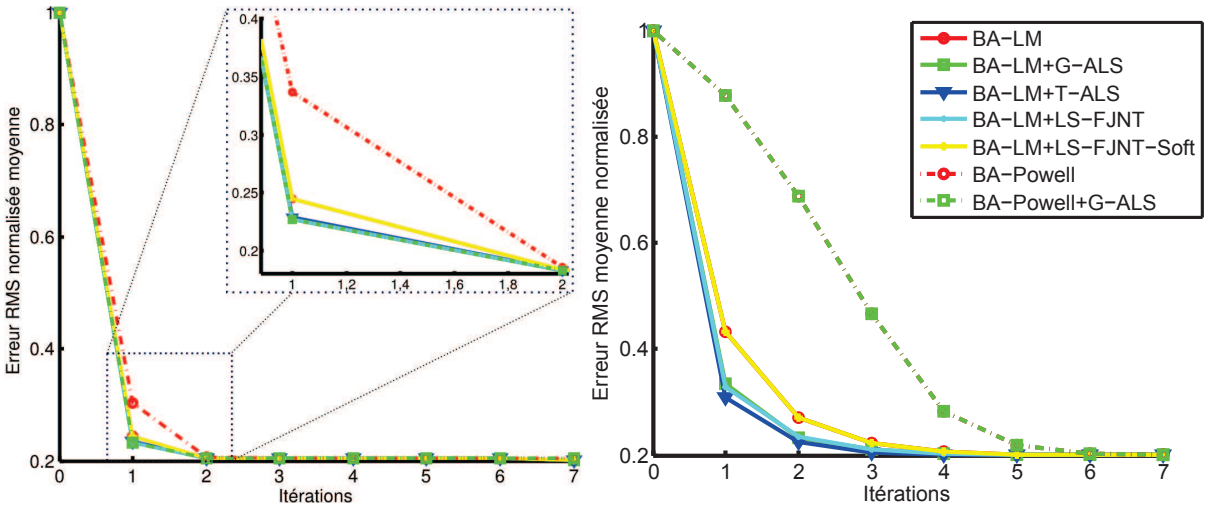
Nous présentons dans cette section les résultats des différents algorithmes d'optimisation non-linéaires pour des ajustement de faisceaux calibrés et non calibrés. La figure 4.5 synthétise les résultats de ces différents algorithmes et présente l'évolution de l'erreur quadratique moyenne normalisée RMS (*Root Mean Square*) : $\bar{\varepsilon}^{(k)} = E[\frac{RMS_{2D}(\mathbf{x}^{(k)})}{RMS_{2D}(\mathbf{x}^{(0)})}]$, au cours des itérations des optimisations. Nous avons généré 40 jeux de données de synthèse, dont le procédé de génération est mentionné dans la section 4.4.1.2, et nous avons ensuite lancé chaque algorithme d'optimisation sur ces jeux, pour les deux types d'ajustement de faisceaux calibré (20) et non calibré (20). L'erreur quadratique moyenne pour chaque type d'algorithme est ainsi estimée sur 20 simulations distinctes.

4.4.2.1 Première expérience

Les deux approches (*G-ALS* et *T-ALS*) ont été intégrées à un algorithme de LEVENBERG-MARQUARDT et correspondent respectivement aux courbes pleines verte (`BA-LM+LS-G-ALS`) et bleue (`BA-LM+LS-T-ALS`). Nous avons de plus testé le couplage de l'algorithme de *Dog-Leg* et de la recherche linéaire algébrique globale (courbe verte en pointillés).

Intéressons nous tout d'abord à la partie de gauche de la figure 4.5, c'est-à-dire dans le cas d'un ajustement de faisceaux calibré. Nous observons dans ces conditions de synthèse que la principale différence entre les algorithmes d'optimisation se situe lors de la première itération de l'optimisation. Alors que les algorithmes dépourvus de recherche linéaire comme `BA-LM` ou `BA-Powell` proposent en moyenne une décroissance de 75% de l'erreur RMS normalisée au maximum (66% pour de `BA-Powell`), les algorithmes dotés d'une recherche linéaire augmentent légèrement cette décroissance, avec en moyenne une décroissance de 78%, soit un gain de 3% par rapport à `BA-LM`. Les deux variantes de recherche linéaire algébrique globale et à deux dimensions améliorent donc bien la minimisation, et apportent en moyenne dans ces conditions de bruit et d'initialisation des paramètres, un gain relativement faible mais néanmoins observable. Elles obtiennent de plus les mêmes résultats que la recherche linéaire exacte `BA-LM+LS-FJNT` de [Frandsen 1999], précise mais très lente. Cela signifie que nous faisons au mieux de ce qu'il est possible de faire dans la direction choisie par l'algorithme de LEVENBERG-MARQUARDT. Notons que l'heuristique *soft* de la recherche linéaire de [Frandsen 1999] n'estime pas, en moyenne, une amplitude suffisamment efficace pour améliorer la convergence de la minimisation. Cela est certainement dû au conditionnement du problème.

Les résultats sur les ajustements de faisceaux non calibrés (figure de droite) montrent que l'accélération apportée par la technique de recherche linéaire, quelle soit algébrique ou non



(a) (b)

FIGURE 4.5 – Convergences moyennes de différents algorithmes d'ajustement de faisceaux sur des ajustements calibrés (a) et non calibrés (b). L'erreur est normalisée par l'erreur RMS initiale et est une moyenne estimée sur 20 simulations.

(`LS−FJNT`) est plus importante dans les ajustements non euclidiens (non calibrés). Cela est semble-t-il dû à l'élargissement de l'espace des paramètres de l'optimisation par rapport à une optimisation dans le cas calibré, donc à moins de contraintes pour converger vers le minimum. Dans ces conditions de bruit, le gain moyen en RMS pour les 4 premières itérations est de 11% pour la recherche linéaire algébrique globale et 15% pour la recherche linéaire algébrique à deux dimensions, avec un écart type de 4% pour les deux méthodes. La variante *Two way-ALS* de la recherche linéaire algébrique est sensiblement plus efficace que la version globale et que la recherche linéaire exacte `LS−FJNT` dans ces conditions, mais rappelons qu'il s'agit ici d'une heuristique qui n'est pas à proprement parler une méthode de recherche linéaire classique car elle estime deux amplitudes distinctes pour deux sous-ensembles des paramètres optimisés (caméras et scène). De plus, nous devons rappeler que le vecteur de paramètres initiaux est formé des vraies valeurs synthétiques auxquelles nous avons ajouté des bruits distinct entre la scène et les caméras, ce qui pourrait favoriser le fait d'avoir deux amplitudes différentes entre ces deux ensembles. Notons enfin la convergence peu efficace proposée par l'algorithme des régions de confiances *Dog-Leg* (`BA−Powell`) sur ces jeux de données synthétiques. Il nous semble que ces résultats peuvent provenir d'une valeur non adaptée du paramètres de confiance, fixé pour tous les tests à $\Gamma^{(0)} = 1.0$, comme l'indiquent les auteurs de [Lourakis 2005].

4.4.2.2 Seconde expérience : initialisation peu précise

Une seconde expérience sur les données synthétiques a été effectuée. Elle s'intéresse à étudier le comportement de l'approche par recherche linéaire, lorsque l'initialisation des paramètres est

peu précise. La figure 4.6 illustre les résultats.

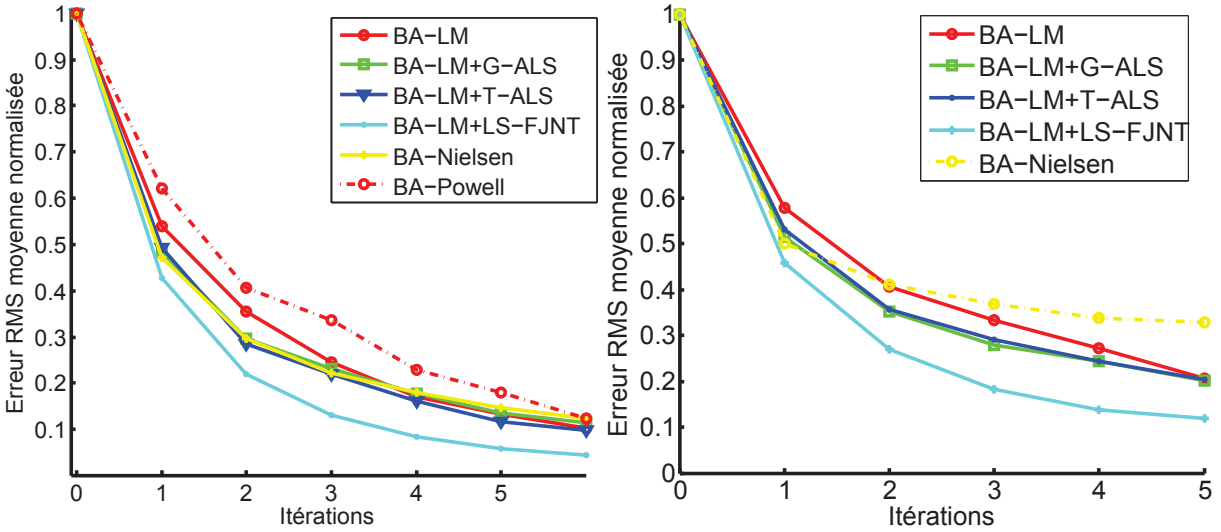


FIGURE 4.6 – Convergences moyennes de différents algorithmes de minimisation non-linéaire sur des ajustements calibrés (a) et non calibrés (b), lorsque l'initialisation des paramètres est peu précise. L'erreur est exprimée en relatif (normalisée par l'erreur RMS initiale) et est une moyenne estimée sur 20 simulations.

En augmentant l'imprécision de l'estimation initiale des paramètres de projection et de scène (respectivement $\sigma_t = 2$ m ($\frac{1}{10}$ème du champ), $\sigma_R = 15°$) et $\sigma_Q = 1.0$ m), l'apport d'une technique de recherche linéaire est plus prononcé que lorsque l'initialisation est près de l'optimum. En effet, nous observons dans ces conditions que la technique de recherche linéaire algébrique globale et sa variante à deux dimensions ont approximativement les mêmes résultats et sont tous deux plus efficaces dans les 4 premières itérations, dans les ajustements de faisceaux calibrés ou non calibrés. Relevons de plus que la recherche linéaire exacte `BA-LM+LS-FJNT` de [Frandsen 1999] surpasse tous les autres algorithmes et converge bien plus rapidement, en nombre d'itérations, mais pas en temps de calcul (voir le tableau 4.1). Notons que tous les algorithmes présentés dans ces graphes convergent en moyenne vers la même solution mais en plus de 5 itérations. Par soucis de lisibilité nous présentons ici que les 5 premières itérations.

Le tableau 4.1 présente les nombres d'itérations et temps moyens réalisés par les différentes méthodes d'optimisation sur le jeu de données de synthèse. Sur ces résultats, seules les optimisations qui ont convergé vers une erreur proche ou inférieure à la valeur obtenue par LEVENBERG-MARQUARDT sans recherche linéaire sont considérées (une grande majorité).

On peut remarquer que les deux techniques de recherche linéaire algébrique diminuent non seulement le nombre total d'itération pour converger vers une même solution (18.8 (*G-ALS*) et 18.3 (*T-ALS*) itérations contre 21.3 pour LEVENBERG-MARQUARDT sans recherche linéaire), mais elles réduisent aussi nettement le temps total de calcul. Un gain de temps d'environ 19% est

Méthode	**Nombre moyen d'itérations**	**Temps moyen de convergence** (s)
`BA-LM`	21.3	166.72
`BA-LM+LS-G-ALS`	18.8	139.53
`BA-LM+LS-T-ALS`	18.3	151.68
`BA-LM+LS-FJNT`	17.2	797.33
`BA-LM+LS-FJNT-soft`	18.6	322.82
`BA-Powell`	21.8	158.97
`BA-Nielsen`	22.8	167.41

TABLE 4.1 – Temps et nombre d'itérations moyens réalisés par les méthodes d'optimisation pour converger sur les données de synthèse.

dans ces conditions d'expérimentation appréciable avec la recherche linéaire algébrique globale. La variante à deux dimensions, apporte une diminution de 10% sur le temps de calcul total de l'optimisation. Bien que cette méthode réalise en moyenne moins d'itérations que la version globale, le temps nécessaire à résoudre le polynôme de degré 5 est plus important qu'avec un polynome de degré 3, ce qui explique en partie la différence de temps. Une autre possibilité est que, nous le verrons, le taux d'acceptation de l'amplitude pour la version *T-ALS* est moins important que pour la méthode *G-ALS*.

Taux d'acceptation des amplitudes. Rappelons que la recherche linéaire algébrique estime une amplitude de déplacement à partir d'une approximation de l'erreur géométrique : l'erreur de reprojection algébrique. Or, puisqu'il s'agit d'une approximation de la fonction objectif originale, il est nécessaire de vérifier que l'hypothèse d'amplitude est effectivement efficace au regard de cette dernière, c'est ce que nous appelons les critères d'acceptation (voir les sections 4.2.3.2 et 4.2.4.2.

Les taux d'acceptation (validant l'ensemble des critères) moyens (estimés sur une cinquantaine de données réelles et simulées) pour chaque itération des optimisations avec recherche linéaire algébrique globales et à deux dimensions sont présentés dans le tableau 4.2. Une représentation graphique comprenant les écarts types est proposée dans la figure 4.7.

Ces résultats indiquent d'une part que les hypothèses d'amplitude sont plus souvent acceptées dans les premières itérations des optimisations pour les deux variantes de recherche linéaire algébrique. Et d'autre part, la variante de recherche linéaire algébrique à deux dimensions possède un taux d'acceptation des hypothèses d'amplitudes inférieur à la version de recherche linéaire algébrique globale. Peut être est-ce parce qu'elle estime deux amplitudes distinctes pour les deux ensembles de paramètres, qui doivent répondre ensemble aux critères d'acceptation.

Nombre et fréquence des hypothèses. Le tableau 4.3 présente le nombre et la fréquence des hypothèses générées par les recherche linéaire algébrique globale (*G-ALS*) et à deux dimensions (*Two way-ALS*), estimés sur de multiples essais sur des données réelles et simulées. Si l'on ob-

	G-ALS	*Two way-ALS*
Itér. k	**Taux d'acceptation** (%)	**Taux d'acceptation** (%)
1	58.66	44.00
2	45.33	37.33
3	37.33	29.33
4	21.33	13.33
5	9.33	8.00
6	12.00	5.33

TABLE 4.2 – Taux d'acceptation moyens des hypothèses *G-ALS* et *T-ALS* suivant les itérations de l'optimisation, sur une cinquantaine de données réelles et simulées.

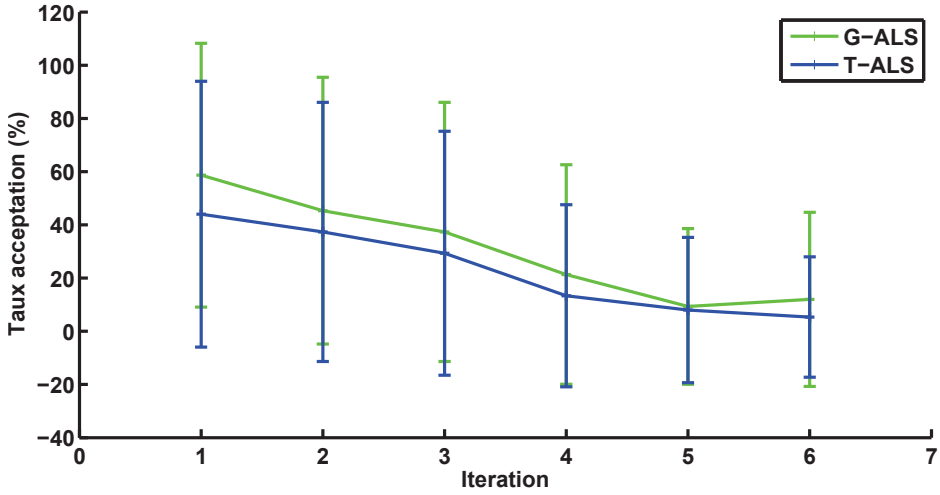


FIGURE 4.7 – Taux d'acceptation moyens et écarts types des hypothèses, estimés sur une cinquantaine de données réelles et simulées.

serve le nombre d'hypothèses proposées par les deux variantes de recherche linéaire algébrique, c'est-à-dire le nombre de racines du polynôme de degré 3 (pour le *G-ALS*) ou 5 (pour le *Two way-ALS*), on observe que dans la majorité des cas une seule hypothèse réelle est générée. En pratique, dans le cadre des ajustement de faisceaux calibré et non calibré effectués sur quelques milliers de variables, les coefficients des polynômes impliquent qu'il est beaucoup plus fréquent d'avoir une seule et unique racine réelle. Il arrive toutefois d'avoir plusieurs hypothèses à tester et, comme les polynômes sont de degré impair (3 et 5), il est rare d'avoir un nombre de racines pair.

Exemples de valeurs d'amplitudes. Nous donnons à titre indicatif quelques valeurs d'amplitudes estimées par les deux variantes de recherche linéaire algébrique ainsi que la validation ou non de l'ensemble des critères pour les 6 premières itérations de l'optimisation. Ces résultats sont synthétisés dans les tableaux 4.4, 4.5 et 4.6 et correspondent au ajustement de faisceaux

	G-ALS		*Two way-ALS*	
Nb. racines	**Nb.**	**Fréq.** (%)	**Nb.**	**Fréq.** (%)
1	438	97.33	440	97.77
2	0	0	0	0
3	9	2.66	7	2.33
4	-	-	0	0
5	-	-	0	0

TABLE 4.3 – Nombre (et fréquence) de racines du polynôme de degré 3 (*G-ALS*) ou 5 (*Two way-ALS*), calculés sur de multiples données réelles et simulées.

calibrés pour respectivement une séquence de synthèse et les séquences réelles `LesUlis30` et `LesUlis6`. Nous pouvons remarquer qu'il est fréquent, mais pas systématique, que les deux recherche linéaire algébrique proposent une amplitude de déplacement qui diminue l'erreur de reprojection. Cela est principalement vrai dans les premières itérations mais arrive quelquefois aussi dans les itérations suivantes. Nous pouvons observer aussi que les hypothèses proposées sont parfois très éloignées de l'amplitude de référence (l'unité), c'est notamment le cas dans les dernières itérations de la minimisation, où les minima des fonctions de coût algébriques et géométriques diffèrent particulièrement.

	G-ALS		*Two way-ALS*		
Itér. k	$\alpha^{(k)}$	Acceptée ?	$\alpha_{\mathbf{P}}^{(k)}$	$\alpha_{\mathbf{Q}}^{(k)}$	Acceptée ?
1	1.117	**oui**	1.304	1.054	non
2	1.411	**oui**	1.423	1.387	**oui**
3	1.166	non	0.923	0.979	non
4	1.711	**oui**	1.890	1.878	**oui**
5	2.390	non	2.916	2.929	non
6	0.037	non	5.771	6.875	non

TABLE 4.4 – Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur un même jeu de données de synthèse.

4.4.3 Résultats sur des données réelles

Nous avons expérimenté notre approche sur des données réelles issues de plusieurs sources : appareils photographiques ou caméras vidéo.

	G-ALS		*Two way-ALS*		
Itér. k	$\alpha^{(k)}$	Acceptée ?	$\alpha_{\mathbf{P}}^{(k)}$	$\alpha_{\mathbf{Q}}^{(k)}$	Acceptée ?
1	1.069	**oui**	1.438	0.957	**oui**
2	1.725	**oui**	1.622	1.804	**oui**
3	0.300	non	0.905	0.337	non
4	−1.174	non	−0.921	−1.148	non
5	−4.367	non	−6.593	−4.160	non
6	−23.819	non	−30.5924	−23.179	non

TABLE 4.5 – Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur le jeu de données `LesUlis30`.

	G-ALS		*Two way-ALS*		
Itér. k	$\alpha^{(k)}$	Acceptée ?	$\alpha_{\mathbf{P}}^{(k)}$	$\alpha_{\mathbf{Q}}^{(k)}$	Acceptée ?
1	1.352	non	2.391	1.099	non
2	2.550	non	2.582	2.552	non
3	0.759	non	0.579	0.725	non
4	−0.154	non	−0.374	−0.148	non
5	−0.766	non	−1.001	−0.762	non
6	−3.172	non	−3.656	−3.194	non

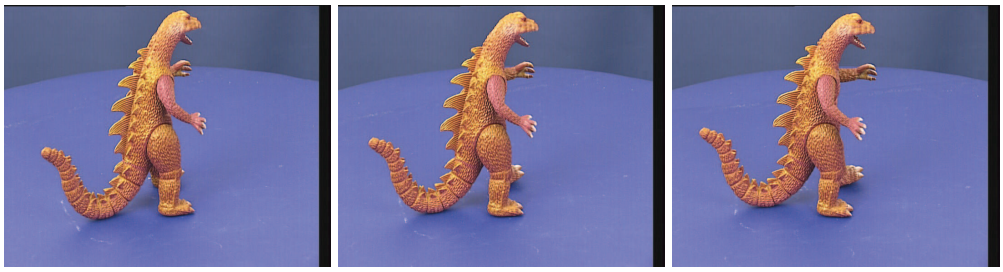
TABLE 4.6 – Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur le jeu de données `LesUlis6`.

4.4.3.1 Données réelles

Nous avons premièrement employé les données `Multiview dataset` issue de la banque de données du *Visual Geometry Group* de l'université d'Oxford [6], et notamment les séquences `Dinosaur`, `Castle` et `House`. Trois images de ces séquences sont reproduites dans les figures 4.8 à 4.10. Elles sont constituées respectivement de 36 images couleurs de 720×576 pixels, de 46 images couleurs de tailles variées et de 10 images en niveaux de gris de tailles 768×576 pixels. Notons que nous n'utilisons pas l'intégralité des images pour les deux premières séquences. Nous sélectionnons seulement les images possédant un nombre important de points 3D communs, appariés correctement, et ce afin de mieux contraindre l'estimation initiale réalisée par factorisation. La méthode étant en effet assez sensible au manque de données.

Le deuxième jeu de données réelles a été obtenu à l'aide une caméra vidéo placée sur un véhicule se déplaçant dans un environnement urbain (Les Ulis, Essonnes) sur 500 mètres environ. Quelque unes des 1000 images constituant la séquence dénommée `LesUlis` sont présentées dans la figure 4.11. Cette dernière séquence sera utilisée pour valider les deux approches de

6. http://www.robots.ox.ac.uk/~vgg/data/data-mview.html



(a) (b) (c)

FIGURE 4.8 – Images issues de la séquence `Dinosaur`.

(a) (b) (c)

FIGURE 4.9 – Images issues de la séquence `Castle`.

recherche linéaire algébrique dans deux applications, la reconstruction 3D par *batch Structure-from-Motion* et par *Structure-from-Motion* incrémental.

4.4.3.2 Résultats avec un algorithme de *batch Structure-from-Motion*

Dans ces expérimentations le calcul initial (précédant l'optimisation) des poses de la caméra et de la structure 3D de la scène a été effectué par un algorithme de *batch Structure-from-Motion* (section 3.3.1 page 86). Ce type d'algorithme délivre souvent une estimation peu précise de la reconstruction, la qualité dépendant en partie de la fréquence d'observation des points 3D de la scène. Dans ces expérimentations, nous avons appliqué les deux approches *G-ALS* et *T-ALS* seulement durant les 5 premières itérations de l'optimisation puisque dans notre contexte la convergence est souvent rapide (quelques itérations) et ne nécessite donc pas de recherche linéaire dans les dernières itérations.

Le tableau 4.7 synthétise l'ensemble des résultats sur les données réelles. Un exemple de convergence en temps et en itérations pour les séquences `LesUlis6` et `LesUlis30` est proposé dans la figure 4.12.

Ces résultats montrent que les deux techniques de recherche linéaire algébrique accélèrent souvent le processus d'optimisation, apportant un gain en temps de calcul de 6 à 20% (sur la séquence `Dinosaur`) par rapport à une technique d'optimisation par LEVENBERG-

(a) (b) (c)

FIGURE 4.10 – Images issues de la séquence `House`.

(a) (b) (c)

FIGURE 4.11 – Images issues de la séquence `LesUlis`

Séquence	N_m	N_p	**RMS Init.** (pixels)	**RMS final** (pixels)			**Temps** (s) **/ Itérations**		
				sans	*G-ALS*	*T-ALS*	sans	*G-ALS*	*T-ALS*
`Dinosaur`	12	4035	6.345	1.534	1.534	1.534	19.058/8	15.872/6	14.750/6
`Castle`	9	3960	5.364	3.121	3.121	3.121	138.02/25	130.15/24	129.83/24
`House`	10	2076	8.189	0.627	0.627	0.627	23.463/14	21.312/13	21.058/13
`LesUlis30`	30	17136	2.642	0.668	0.668	0.668	39.93/8	34.98/7	35.17/7
`LesUlis6`	6	2466	0.329	0.281	0.281	0.281	8.83/4	10.13/4	9.84/4

TABLE 4.7 – Résultats des ajustements de faisceaux calibrés par la méthode LEVENBERG-MARQUARDT sur les séquences réelles, sans et avec la recherche linéaire algébrique (globale *G-ALS* et à deux dimensions *T-ALS*).

MARQUARDT. Cette diminution du temps global nécessaire à l'optimisation pour converger vers une solution stable peut être expliquée en comparant le nombre d'itérations que chaque méthode réalise. En effet, les heuristiques nécessitent moins d'itérations pour converger et, même si elles ajoutent un temps de calcul supplémentaire, un gain de temps est observable. Dans la figure 4.12(a) (courbe du bas), nous observons à la deuxième itération que l'algorithme de LEVENBERG-MARQUARDT dépourvu de recherche linéaire n'accepte pas le déplacement et doit donc modifier son paramètre de régularisation, mais avec les deux approches ainsi qu'avec la

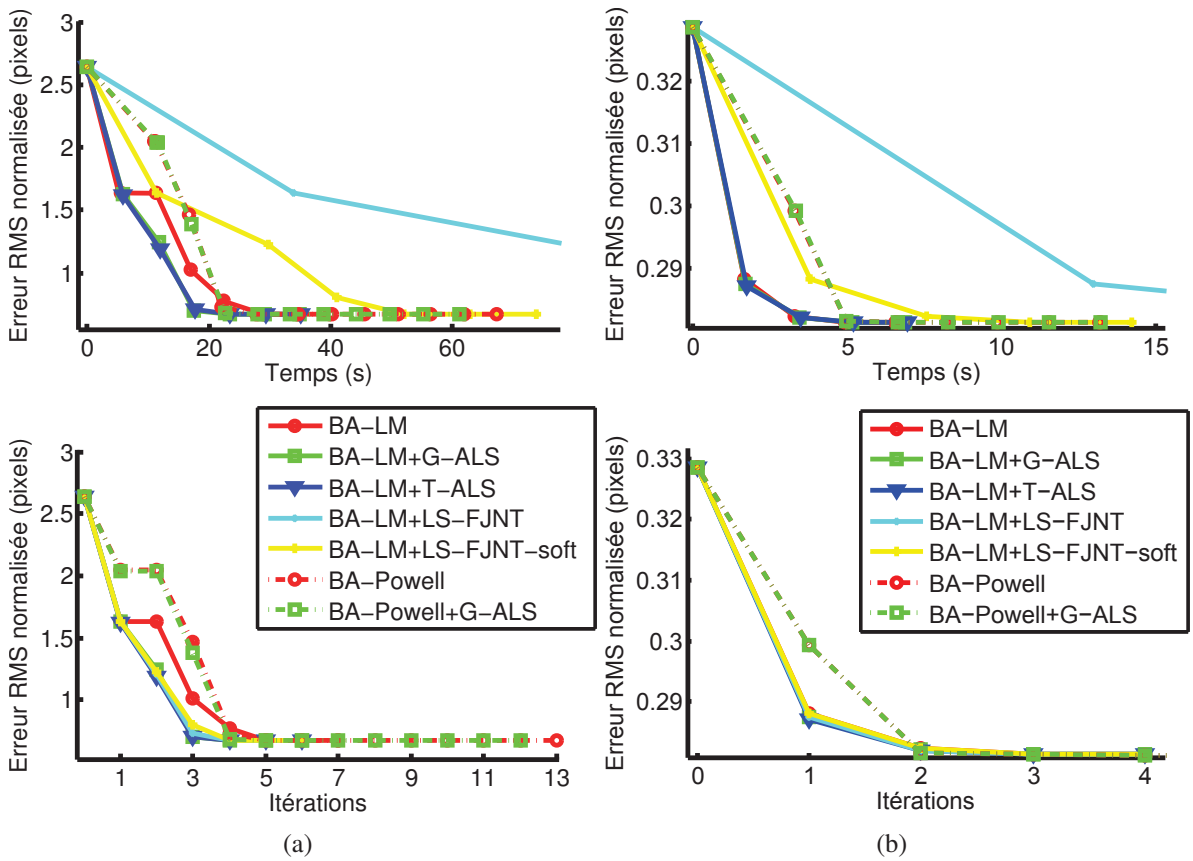


FIGURE 4.12 – Convergence des ajustement de faisceaux sur les séquences `LesUlis30` (a, 30 caméras) et `LesUlis6` (b, 6 caméras). Les courbes en haut représentent l'évolution du RMS au cours du temps et les courbes situées en bas représentent l'évolution du RMS suivant les itérations de l'optimisation.

recherche linéaire `FJNT` de [Frandsen 1999] l'itération est acceptée grâce à l'estimation d'une amplitude efficace pour le déplacement. Quand la méthode `FJNT` de [Frandsen 1999] prend une trentaine de seconde pour estimer cette amplitude, les deux variantes de recherche linéaire algébrique mettent très peu de temps (quelques secondes). Remarquons que l'algorithme de *Dog-Leg* est aussi sujet à ce problème de l'itération 2, mais consomme beaucoup moins de temps CPU pour changer de comportement.

La variante à deux dimensions de la recherche linéaire algébrique semble apporter une légère amélioration par rapport à la recherche linéaire algébrique globale dans ce contexte, ce qui confirme les résultats obtenus sur les données synthétiques.

Il faut toutefois préciser que dans certaines conditions, la méthode de recherche linéaire algébrique proposée ne semble pas améliorer la convergence, et augmente de plus le temps de calcul de l'optimisation. C'est notamment le cas de la séquence `LesUlis6`. Dans cette séquence, remarquons que l'erreur initiale avant l'optimisation est déjà inférieure au pixel (0.329 pixels),

or ceci correspond au cas où l'algorithme de LEVENBERG-MARQUARDT adopte la convergence quadratique du GAUSS-NEWTON et ne nécessite pas de recherche linéaire.

4.4.3.3 Résultats avec un algorithme de *Structure-from-Motion* incrémental

Nous avons de plus expérimenté les techniques de recherche linéaire algébrique dans les conditions d'un algorithme de *Structure-from-Motion* incrémental. Nous comparons principalement ici le comportement des deux techniques sur la séquence `LesUlis`, dans deux ajustement de faisceaux calibrés local (15 caméras) et global (150 caméras).

Un exemple de reconstruction 3D après l'ajustement de faisceaux est présenté en figure 4.13.

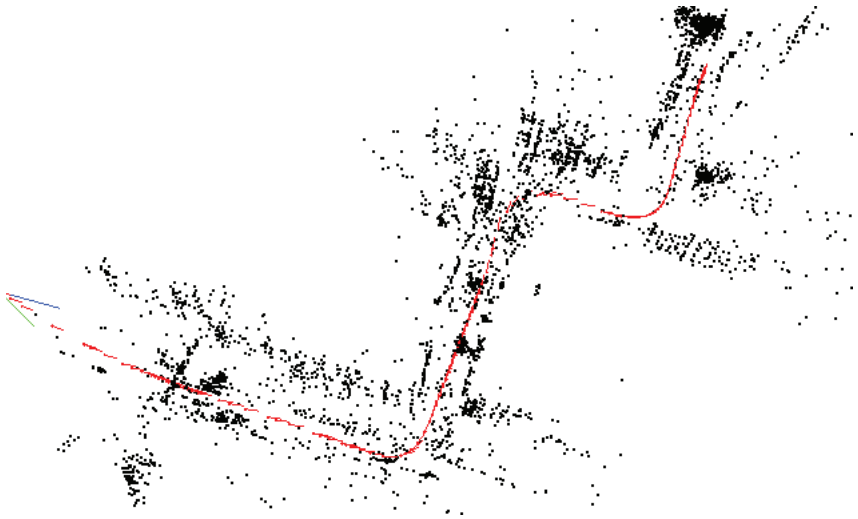


FIGURE 4.13 – Reconstruction 3D de la ville et localisation du véhicule pour la séquence `les ulis` 150 poses.

Les différents résultats sont présentés dans la figure 4.14.

Nous pouvons remarquer dans la figure 4.14(a) que dans le cas de l'ajustement de faisceaux local, les deux techniques de recherche linéaire algébrique n'apportent quasiment aucune amélioration à l'optimisation (excepté pour le *G-ALS* à l'itération 4). En effet, rappelons que dans les ajustements de faisceaux locaux, l'erreur de reprojection est raffinée au fur et à mesure des images clés. Or, l'algorithme LEVENBERG-MARQUARDT acquière dans ces conditions un comportement de type GAUSS-NEWTON puisque l'état courant est périodiquement rapproché du minimum (ajustement de faisceaux local) et n'est donc jamais très loin de celui-ci. Comme nous l'évoquons dans la section 2.2.2.2, la méthode GAUSS-NEWTON propose intrinsèquement une amplitude de déplacement déjà très efficace, d'autant plus lorsque l'approximation de la matrice hessienne est précise. Nous pouvons donc conclure que dans ce type de raffinement local, notre technique de recherche linéaire algébrique, comme d'ailleurs l'ensemble des techniques de recherche linéaire ne semblent pas nécessaires et ne font pas accélérer l'optimisation. L'efficacité

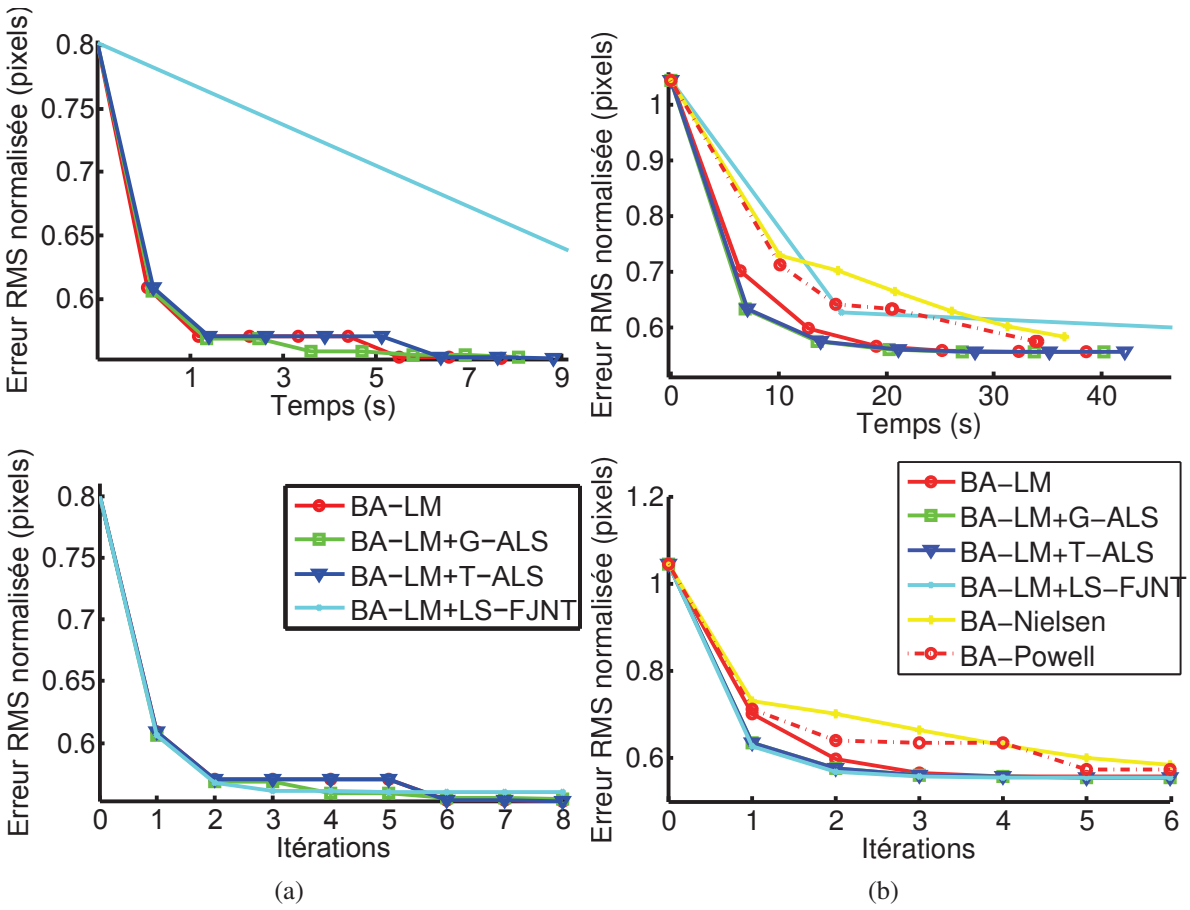


FIGURE 4.14 – Évolution du RMS (pixels) par rapport au temps de calcul (en haut) et par rapport aux itérations (en bas), pour des ajustements locaux (a) et globaux (b).

de l'ALS sera d'autant plus importante que l'algorithme LEVENBERG-MARQUARDT choisira un comportement d'une descente de gradient.

À l'inverse, lorsque l'on considère des ajustements globaux (résultats présentés sur les 6 premières itérations en figure 4.14(b)) et comparons les deux variantes de recherche linéaire algébrique (courbes verte et bleue) par rapport à la méthode LEVENBERG-MARQUARDT sans recherche linéaire (courbe rouge pleine), celles-ci apportent réellement un gain de temps à l'optimisation. Les temps de convergence des trois premières méthodes (`BA−LM`, `BA−LM+LS−G−ALS` et `BA−LM+LS−T−ALS`) lorsque l'on procède à toutes les itérations sont respectivement de 84.21, 79.33 et 83.32 secondes en 13, 12 et 12 itérations. La minimisation converge plus rapidement notamment dans les premières itérations et permet de réaliser moins d'itération que l'algorithme classique `BA−LM`. Les heuristiques *G-ALS* et *T-ALS* possèdent de plus souvent le même comportement que le `BA−LM+LS−FJNT` notamment dans les ajustements métriques, mais sont bien plus rapides.

4.5 Discussion et conclusion

Nous avons présenté dans ce chapitre une nouvelle méthode de recherche linéaire adaptée aux problèmes de reconstruction à partir du mouvement. L'idée principale consiste à utiliser une fonction objective basée sur la distance algébrique seulement dans le cadre d'une recherche linéaire, c'est-à-dire seulement dans le but d'estimer l'amplitude des déplacements dans l'espace des paramètres.

Une méthode générique. Cette technique de recherche linéaire algébrique est générique car elle peut être utilisée dans de nombreux problèmes non-linéaires, dans lesquels la fonction objectif est constituée de résidus, de distances entre des points 2D par exemple, mais elle semble aussi être utilisable sur des distances entre des points de degré supérieur, (3D ou plus). Notre proposition est aussi générique dans le sens où elle peut être employée avec différents algorithmes d'optimisation non-linéaire. Nous avons ici choisi d'étudier l'apport de la recherche linéaire algébrique couplée à un algorithme de LEVENBERG-MARQUARDT car il s'agit de l'algorithme le plus employé en vision par ordinateur et l'un des plus efficace. Il peut être envisageable aussi de coupler notre approche avec la technique des gradients conjugués avec préconditionnement du problème.

Une recherche linéaire algébrique efficace et adaptée à l'ajustement de faisceaux. Dans cette thèse nous avons étudié l'apport de cette technique dans un ajustement de faisceaux et avons remarqué l'intérêt de celle-ci, notamment dans les problèmes dont l'initialisation est peu précise, c'est-à-dire généralement dans les problèmes à grand nombre de variables. La méthode n'améliore pas la convergence dans tous les cas mais, en moyenne, une accélération de la convergence est observable sur les données synthétiques comme réelles, notamment dans les premières itérations de l'optimisation. C'est pourquoi nous conseillons d'utiliser l'approche proposée seulement dans les n premières itérations. Le nombre dépendra bien entendu de l'application ciblée mais dans notre contexte d'ajustement de faisceaux avec quelques milliers de variables optimisées simultanément, $n = 5$ itérations semblent suffisantes. Nous pensons que la méthode n'est en revanche pas adaptée à l'application d'un SLAM incrémental (par ajustement de faisceaux local) dans laquelle l'optimisation est presque toujours près de l'optimum. Il semblerait intéressant de tester la méthode dans le cadre d'un ajustement de faisceaux à grande échelle (*large scale*) car *a priori* la recherche linéaire algébrique semble bien adaptée. Nous pouvons aussi imaginer un critère sur la valeur du paramètre de régularisation λ, par exemple, ne lancer la recherche linéaire algébrique que si celui-ci est supérieur à un seuil $\lambda > \lambda_{ALS}$ ou simplement lorsque l'algorithme LEVENBERG-MARQUARDT augmente ce paramètre à cause d'une mauvaise approximation de la matrice hessienne.

Le cas de la variante à deux dimensions. Dans le problème d'ajustement de faisceaux, il n'est pas rare que la qualité de l'estimation initiale de la structure de la scène soit moins précise que l'estimation initiale des matrices de projection. Dans ce cas, l'heuristique de recherche linéaire

algébrique à deux dimensions peut être employée. Le gain en temps de calcul n'est cependant pas très important au regard des résultats obtenus avec l'approche *G-ALS*, d'autant que nous faisons appel à la fonction `roots` de Matlab pour déterminer les racines du polynôme de degré 5. Cette fonction est optimisée (écrite en C ou C++) et le temps nécessaire à cette estimation n'apparaît que très peu dans les résultats (en comparaison avec le reste des calculs effectué en langage Matlab).

La recherche linéaire algébrique pour éviter les minima locaux. Notons de plus que, en théorie, la recherche linéaire algébrique pourrait permettre d'éviter certains minima locaux de la fonction objectif étant donné que nous employons une approximation quadratique différente de celle de TAYLOR, qui peut ne pas avoir les mêmes minima. Cette approximation donne la possibilité grâce à la résolution polynomiale de proposer une amplitude de déplacement qui va au delà de l'approximation locale de TAYLOR. L'acceptation ou non de cette hypothèse d'amplitude qui peut être lointaine et pourquoi pas négative, reviendra alors aux différents critères d'amélioration. Nous pensons qu'une étude plus importante sur la capacité de la recherche linéaire algébrique à éviter les minima locaux serait appréciable.

Les critères d'amélioration. Les critères d'amélioration, comprenant notamment le premier critère de WOLFE, permettent non seulement de garantir une certaine décroissance de la fonction objectif dans le choix de l'amplitude, mais permettent aussi de sélectionner l'amplitude la plus efficace lorsque plusieurs hypothèses sont estimées par la recherche linéaire algébrique. En pratique, nous avons cependant observé que le nombre d'hypothèses était souvent faible et généralement une seule hypothèse d'amplitude est proposée par les deux variantes de la recherche linéaire algébrique dans l'application d'un ajustement de faisceaux.

CHAPITRE 5

Fusion de données par ajustement de faisceaux

Ce chapitre introduit une méthode permettant d'intégrer des données de localisation issues d'un second capteur, dans une application de localisation par vision, lorsque les incertitudes des contraintes sont inconnues ou imprécises. Ces données sont intégrées dans l'ajustement de faisceaux sous la forme de contraintes de régularisation pondérées par un coefficient. Nous étudions différentes techniques permettant de sélectionner dynamiquement les coefficients de pondération associées aux contraintes. Enfin, nous proposons d'employer cette technique dans le cadre d'un algorithme de localisation multi-capteur dans lequel un second capteur est utilisé afin de contraindre la trajectoire estimée par l'odométrie visuelle. Ces travaux de recherche ont été publiés dans la conférence internationale 3DPVT 2010 [Michot 2010a] et un brevet a été déposé [Michot 2010b] (brevet en cours de révision).

This chapter introduces a method for integrating location data from a motion sensor, in a vision based tracking system, when the uncertainties of the measurements are unknown or unclear. These data are integrated into the local ajustement de faisceaux of a visual SLAM as regularization constraints. We study three different techniques that dynamically select the weights associated to the constraints. Finally, we validate the use of this technique as part of a multi-sensor localization algorithm in which a motion sensor is used together with a camera to constrain the trajectory estimated by visual odometry for two applications : vehicle and person localization. This research were published in the International Conference 3DPVT 2010 [Michot 2010a] and a (pending) patent has been filed [Michot 2010b].

5.1 État de l'art

5.1.1 Problématique

Les systèmes de localisation et reconstruction 3D par vision monoculaire sont aujourd'hui activement employés dans de multiples applications : *match moving* (logiciels `Boujou` et `MatchMover`), localisation en milieu inconnu par SLAM [Davison 2003, Mouragnon 2006], navigation autonome, réalité augmentée [Klein 2007], etc. L'un des principal avantages des caméras par rapport aux centrales de navigation (INS) par exemple pour localiser un système, est leur capacité à fournir de nombreuses informations redondantes sur la scène observée et, par inférence, sur le déplacement du système.

Mais ces techniques de localisation basées uniquement sur la vision peuvent être sensibles à un certain nombre de problèmes : impossibilité d'estimation à cause de mauvaises conditions visuelles (occultation, faible quantité de texture, faible luminosité, etc.), dérives (en position, en orientation ou en échelle) dues aux bruits, à la géométrie de la scène, aux approximations des modèles ainsi qu'aux minima locaux liés aux algorithmes d'optimisation non-linéaire. Pour palier ces difficultés, il est courant de combiner des informations complémentaires provenant de différentes sources : de caméras supplémentaires (stéréo [Howard 2008], multi-caméra [Kaess 2009]), de capteurs de mouvement (IMU [You 2001, Huster 2003, Strelow 2004, Hol 2006, Oskiper 2007, Armesto 2007, Gemeiner 2007, Hol 2007, Servant 2010], odomètre [Chenavier 1992, Eudes 2010a],...) ou de position (INS, GPS [Agrawal 2006b], ...), de capteurs mesurant l'environnement (laser LiDAR [Newman 2006, Mirisola 2007, Craciun 2008, Naikal 2009], sonar [Ribas 2006], etc.) ou par l'utilisation d'informations ou *a priori* issues d'autres méthodes (reconnaissance de lieu [Eade 2008, Angeli 2008], contraintes géométriques sur la scène [Lothe 2009, Lothe 2010] ou sur le déplacement, etc.). Ces observations redondantes informent directement ou indirectement de la trajectoire du système ou de la géométrie de la scène.

Pour résoudre le problème inverse dans lequel on souhaite estimer la trajectoire du système et la structure de la scène à partir de l'ensemble des observations, le problème est généralement exprimé sous la forme d'un filtrage Bayésien et résolu par l'intermédiaire d'estimateurs au maximum de vraisemblance ou au maximum *a posteriori* (MAP, voir la section 2.1). Ces méthodes recherchent alors la localisation du système qui justifie au mieux l'ensemble des observations collectées. L'approche classique consiste à faire une hypothèse sur la distribution du bruit dans les observations, et considérer que chaque bruit suit une loi normale centrée de variance σ^2 : $z = \mathcal{N}(0, \sigma^2)$. Lorsque les fonctions de densité des bruits des mesures suivent une loi gaussienne, le problème inverse peut être exprimé comme un problème d'optimisation non-linéaire au sens des moindres carrés (voir la section 2.2.2).

Cependant, les méthodes de l'état de l'art ont été conçues avec l'idée que ces incertitudes du premier ordre, les variances σ_o^2, étaient *a priori* connues. Or, en pratique, il est fréquent que l'information d'incertitude sur les observations n'est pas connue ou imprécise, par exemple pour les données de localisation du GPS. L'estimation de celle-ci peut être parfois trop coûteuse en temps de calcul pour pouvoir être estimées en temps réel. La qualité du calibrage des capteurs

ainsi que l'évolution temporelle de ces paramètres peuvent aussi induire une erreur importante sur l'incertitude réelle des observations. Par exemple, les mesures données par un odomètre dépendent de la pression des pneumatiques du véhicule et des glissements des roues sur la voie, l'information d'incertitude est dans ces conditions peu fiable.

5.1.2 Limites des méthodes existantes

Dans cette section nous présentons les méthodes classiques de fusion de données couramment employées en vision par ordinateur, et plus particulièrement les méthodes adaptées à la localisation en temps réel intégrant une caméra.

5.1.2.1 Fusion de données par filtrage de Kalman et variantes

Le filtre Bayesien le plus employé dans les systèmes de fusion multi-source est certainement le filtre de Kalman et ses diverses variantes [Davison 2003, Huster 2003, Strelow 2004, Montiel 2006, Armesto 2007, Gemeiner 2007, Hol 2007, Civera 2009, Servant 2010]. L'un des principaux avantages de ce filtre est sa capacité à gérer intrinsèquement les variances des bruits des observations et à propager celles-ci naturellement aux paramètres d'état (à la localisation). La communauté de robotique a été très productive dans la recherche sur le SLAM multi-capteur basé sur un filtre de Kalman et ses variantes, voir notamment l'état de l'art effectué à ce sujet par STRELOW [Strelow 2004]. Il existe différentes versions du filtre de Kalman développé à l'origine pour des fonctions de prédiction et d'observation linéaires, puis étendu aux fonctions non-linéaires (EKF : *Extended Kalman Filter*), qui utilisent une approximation locale (de TAYLOR) de celles-ci et qui réalisent une seule itération non-linéaire ou plusieurs (IEKF : *Iterated Extended Kalman Filter*). Le filtre informatif est aussi une variante activement employée [Liu 2003]. Notons aussi l'existence de techniques de fusion de données sans hypothèse sur la distribution du bruit : ce sont les méthodes par filtrage particulaire, pour lesquelles la densité de probabilité des paramètres estimés est représentée par un ensemble de particules [Montemerlo 2002, Montemerlo 2003, Eade 2006].

Alors que le SLAM multi-capteur a montré son efficacité dans les systèmes de localisation et reconstruction, le cas du SLAM visuel temps réel par filtre de Kalman [Davison 2003] restait encore peu performant. Les temps de calcul nécessaires à l'exécution du filtrage par Kalman étendu conduisaient souvent les auteurs à réduire le nombre de points d'intérêt 3D simultanément reconstruits, ce qui avait pour effet de diminuer la précision et la robustesse de la localisation.

Récemment, les importantes avancées apportées à la techniques de l'ajustement de faisceaux (décrit en section 3.2), et notamment sa résolution creuse par blocs et l'introduction du concept d'ajustement de faisceaux local (sur les dernières images clés) [Mouragnon 2006], on fait aujourd'hui de l'ajustement de faisceaux un candidat idéal pour la localisation temps réel par vision. Bien que l'ajustement de faisceaux, sous sa forme générale, est bien plus lent qu'une méthode EKF (elle conserve l'historique et effectue plusieurs itération de GAUSS-NEWTON), il améliorent significativement la précision de la localisation, et peut être employé dans les conditions d'un SLAM temps réel [Mouragnon 2006].

5.1.2.2 Fusion de données par couplage filtre de Kalman-ajustement de faisceaux

Antérieurement aux travaux menés sur le SLAM temps réel par ajustement de faisceaux proposé notamment par MOURAGNON *et al.* [Mouragnon 2006], les recherches se sont orientées vers le couplage de la fusion par filtre de Kalman et l'ajustement de faisceaux. Ceci, afin de bénéficier de la précision de la localisation par vision apportée par l'ajustement de faisceaux, et de la gestion intrinsèque de la fusion multi-capteur par le filtre de Kalman.

C'est notamment ce que proposent STRELOW *et al.* dans [Strelow 2004]. Les auteurs présentent deux algorithmes (hors ligne et en ligne) dans lesquels un calcul de pose est opéré par vision, raffiné par un ajustement de faisceaux, puis fusionné avec les mesures d'orientation provenant d'une centrale inertielle bon marché par un filtre de Kalman étendu itéré (IEKF). Les auteurs montrent que le couplage de l'information image et inertielle avec une centrale inertielle de faible qualité améliore la précision de la localisation.

Des travaux similaires ont été effectués par KONOLIGE *et al.* [Konolige 2007], mais avec un SLAM visuel avec détection des images clés et optimisation par ajustement de faisceaux local. La fusion des données du SLAM visuel et d'une centrale inertielle (ou un GPS [Agrawal 2006b]) est quant à elle opérée *a posteriori* par un filtre de Kalman étendu. La méthode proposée ne s'attache cependant pas à corriger les dérives du SLAM visuel puisque le retour de la fusion de données n'est pas intégré dans le SLAM.

Les travaux de notre thèse s'intéressent aussi à coupler les informations complémentaires (images et inertielles) mais sans utiliser le filtre de Kalman, par un ajustement de faisceaux.

5.1.2.3 Fusion de données par ajustement de faisceaux

Le problème de la fusion de données peut être résolu par la seule utilisation d'un ajustement de faisceaux. En effet, lorsque les bruits des mesures suivent une loi normale centrée de variance connue, alors l'ajustement de faisceaux est un estimateur MAP optimal. Nous différencions les cas où les variances des mesures sont déterminées et précises (par calibrage, valeurs fournies par le constructeur, etc.) ou indéterminées.

Variances des mesures déterminées. Lorsque les variances des mesures sont connues, l'approche de fusion classique et optimale employée est un algorithme d'ajustement de faisceaux avec des termes d'erreurs formé de distances de Mahalanobis [Agrawal 2006a, Predmore 2010]. Nous différencions les applications de localisation hors ligne des applications nécessitant d'être exécutées en temps réel. Dans cette dernière catégorie, il est en général difficile voire impossible d'obtenir la trajectoire optimale de plusieurs sources de données en temps réel puisque cela nécessite beaucoup de temps de calcul afin de mettre à jour l'ensemble de l'historique des poses du système.

Méthodes hors lignes. JUNG et TAYLOR [Jung 2001] présentent une méthode *batch* (hors ligne) de fusion de données vision et inertielles par *Shape-from-Motion*. Ils proposent d'estimer

la meilleure *spline* interpolant la trajectoire des images clés, qui explique au mieux les observations inertielles. Les données ne sont pas fusionnées directement par un algorithme d'optimisation non-linéaire. STRELOW et SINGH [Strelow 2002] proposent aussi une méthode hors ligne de fusion de données vision et inertielles mais en utilisant des optimisations par LEVENBERG-MARQUARDT. La méthode est très précise mais ne peut être simplement étendue aux méthodes de localisation temps réel en raison notamment de la non propagation des distributions de probabilités dans le SLAM par ajustement de faisceaux local. SMITH *et al.* [Smith 2003] s'intéressent à la fusion hors ligne de la trajectoire d'une caméra avec des informations absolues (points de contrôle, GPS). Les informations sont fusionnées de manière optimales par un ajustement de faisceaux en 3 étapes. La première étape consiste à estimer la trajectoire par les algorithmes de vision en commençant par le début de la séquence, de propager les covariances et d'intégrer les points de contrôle par combinaison des erreurs. La deuxième étape est identique à la première mais en partant de la fin de la séquence. Une optimisation finale est ensuite appliquée en minimisant une fonction de coût qui combine l'ensemble des observations et des points de contrôle et les covariances associées. Les auteurs de [Saurer 2010] présentent un système similaire à [Smith 2003], mais s'intéressent à résoudre le problème de fermeture de boucles (*loop closure*) apporté soit par une étape de reconnaissance de lieux, soit par l'utilisateur.

Méthode temps réel. Récemment, avec l'avènement du SLAM visuel temps réel utilisant par exemple l'ajustement de faisceaux local proposé par MOURAGNON *et al.* [Mouragnon 2006], quelques travaux de recherche se sont intéressés à intégrer directement dans l'ajustement de faisceaux des informations extérieures afin de bénéficier des avantages de l'ajustement de faisceaux par rapport au filtre de Kalman [Strasdat 2010]. La principale difficulté est que le l'ajustement de faisceaux local [Mouragnon 2006] ne conserve pas l'incertitude globale de la localisation, ce qui est nécessaire à la fusion précise multi-capteur. En effet, contrairement au filtre de Kalman, l'ajustement de faisceaux ne s'attache pas à propager l'incertitude des observations sur les paramètres optimisés : la matrice de variance-covariance de ces derniers n'est pas calculée de manière explicite. Il est possible d'estimer ces incertitudes en inversant la matrice hessienne du problème, mais dans le cas de l'ajustement de faisceaux local, les incertitudes des caméras non-optimisées ne sont pas propagées aux caméras optimisées, si bien que les incertitudes estimées sont trop optimistes et ne peuvent être efficacement utilisées pour fusionner des informations autre que visuelles. Les travaux récents de EUDES *et al.* [Eudes 2009] sur la propagation d'erreur dans l'ajustement de faisceaux local dans un SLAM monoculaire, ont permis aux mêmes auteurs de réaliser une fusion vision-odomètre dans un ajustement de faisceaux local pondéré [Eudes 2010b] en ajoutant un terme de contrainte sur la norme du déplacement inter-caméras clés dans la fonction de coût. Cette technique de fusion temps réel corrige efficacement mais pas intégralement la dérive d'échelle.

Les méthodes présentées reposent sur le fait que les covariances des observations de tous les capteurs sont connus *a priori*. Mais dans le cas où ces informations d'erreur sont inaccessibles, imprécises ou évoluent au cours du temps de manière indéterminée, les solutions proposées seront moins performantes et nous pensons que l'estimation dynamique du coefficient de

pondération (image de l'isovariance de la contrainte) peut être bénéfique à la localisation.

Variances des mesures indéterminées. Lorsque les variances des observations sont indéterminées, il est toujours possible d'employer l'ajustement de faisceaux afin de fusionner les mesures issues de différentes sources, en combinant tous les termes d'erreur et en pondérant chacun des termes par un facteur, noté λ, qui représente l'importance accordé au terme. Cette forme d'optimisation multiobjectif est récurrente dans de nombreux domaines où il est nécessaire de minimiser conjointement plusieurs termes d'erreur : le recalage (rigide) d'images [Modersitzki 2004] ou (non rigide) de surfaces [Chui 2003, Bartoli 2008], la construction d'image super-résolution [Zibetti 2008] ou plus généralement la régularisation explicite de problèmes mal posés [Hansen 2001], pour réduire la complexité du modèle ou ajouter des *a priori* au modèle.

Tout l'enjeu est ensuite de choisir les valeurs des coefficients de pondération associés aux termes d'erreur ajoutés à la fonction de coût de l'ajustement de faisceaux. Dans la plupart des cas, ces coefficients sont considérés égaux et constants, et sont estimés par calibrage ou empiriquement par tâtonnement. Du point de vue statistique, ces termes de pondération correspondent à l'inverse des isovariances associées aux résidus additionnels. Certains auteurs [Schneider 2007] proposent d'estimer par calibrage (*Variance Component Estimation*) les variances associées à chaque groupe de mesure de plusieurs capteurs, dans le cadre d'une fusion multi-capteur, puis de réaliser un ajustement de faisceaux avec l'ensemble des termes pondérés par les isovariances estimées. Cette méthode est particulièrement efficace et précise lorsque les incertitudes liées aux observations des capteurs sont stables et n'évoluent pas au cours du temps. Elle est cependant rigide dans les autres cas : les poids estimés dépendent alors fortement des données exploitées durant le calibrage et la méthode s'adapte difficilement au fait que la qualité des observations puissent varier au courant du temps.

Il existe des méthodes génériques permettant de sélectionner dynamiquement les coefficients de pondération pour des contraintes de régularisation. Ces méthodes consistent à minimiser un critère mesurant la qualité du modèle, de la localisation, en fonction des coefficients. La validation croisée (voir la section 2.4.1) mesure par exemple la capacité du modèle à prédire de nouvelles données. Elle est aujourd'hui employée afin de sélectionner le poids des contraintes de régularisation [Bartoli 2008] ou des contraintes de lissage de trajectoire dans le cadre d'un SLAM [Farenzena 2008]. D'autres méthodes s'intéressent plus spécifiquement à la régularisation de problèmes génériques mal posés, comme les méthodes *L-Curve* [Hansen 2001], *U-Curve* [Krawczyk-StańDo 2007], *L-Tangent Norm* [Brunet 2008], la méthode de REGIŃSKA [Regiǹska 2002, Bazán 2009], etc. Ces techniques n'ont cependant pas été employées dans le cadre de la fusion multi-capteur.

5.1.3 Positionnement

Comme nous venons de le voir, la plupart des applications de localisation temps réel utilisant plusieurs capteurs fusionnent généralement toutes les informations par une technique de filtrage (par Kalman et ses variantes). Ce n'est que très récemment que certains auteurs [Eudes 2010b]

proposent d'utiliser l'ajustement de faisceaux local afin de fusionner différentes informations provenant de différentes sources. Nous avons aussi choisi d'employer cette dernière approche par ajustement de faisceaux (local) puisque celle-ci est reconnue [Strasdat 2010] pour être la plus efficace, rapide et précise. Ces deux types de méthodes, par filtre de Kalman ou par ajustement de faisceaux, nécessitent cependant de connaître parfaitement les incertitudes (du premier ordre) liées aux observations de chaque capteur pour pouvoir estimer de manière optimale (à l'hypothèse de distribution des erreurs près) la solution du problème inverse. La figure 5.1 représente le schéma de fonctionnement général du SLAM monoculaire contraint par un second capteur.

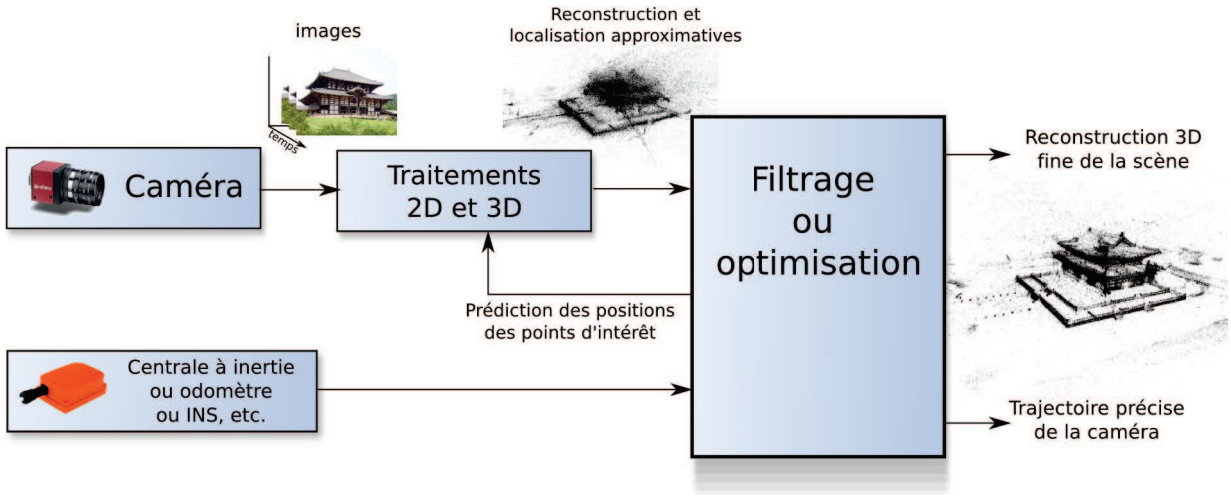


FIGURE 5.1 – Schéma de fonctionnement général du SLAM monoculaire contraint par un second capteur : les mesures du capteur sont intégrées dans l'ajustement de faisceaux.

Dans cette thèse, nous partons du postulat que les informations d'incertitude des mesures sont soit inaccessibles, soit pas assez précises pour être utilisées en toute confiance. Nous conservons l'hypothèse de la distribution normale centrée des bruits, sans que les variances des observations ne soient connues.

Dans ces conditions particulières, nous proposons de résoudre ce problème de fusion de données sans incertitude de façon général, en intégrant à un ajustement de faisceaux des termes de régularisation issus des observations d'un second capteur. La pondération classique par la variance est ainsi remplacée par un paramètre de pondération qui, contrairement aux approches classiques fixant sa valeur empiriquement (par essai-erreur) en obtenant des résultats très différents suivant les valeurs choisies, la pondération sera estimée dans notre cas dynamiquement par différents procédés issus de la sélection de modèle. Des travaux similaires ont été présentés dans [Farenzena 2008], mais les auteurs s'intéressent principalement au problème de lissage de la trajectoire afin de réduire les ambiguïtés dues aux mouvements critiques de la caméra et ne proposent pas de solution pour intégrer ce lissage dans l'ajustement de faisceaux local. Nous proposons de généraliser cette approche dans le cadre de la fusion de données hétérogènes et étudions différentes techniques de sélection de paramètre de pondération.

Notre proposition peut être adoptée afin d'intégrer dans un ajustement de faisceaux local des informations de mouvement, de position, d'échelle ou d'orientation, provenant de capteurs supplémentaires, d'*a priori* ou d'autres algorithmes. Nous présentons son application dans un SLAM multi-capteur.

5.2 Fusion de données d'incertitude inconnue

Nous présentons dans cette section une méthode permettant d'intégrer des contraintes dans un ajustement de faisceaux, lorsque les incertitudes des contraintes ne sont pas connues. Nous présentons différentes techniques permettant de sélectionner dynamiquement les coefficients de pondération qui se substituent aux incertitudes des contraintes.

5.2.1 Contexte et exemple de contraintes

5.2.1.1 Contexte

La méthode de fusion de données générique que nous proposons s'inscrit dans le cadre d'une application de *Structure-from-Motion* séquentiel monoculaire, dans laquelle chaque pose de la caméra est estimée de manière incrémentale, dans l'ordre chronologique d'acquisition. Nous supposons l'existence d'une seconde source k (capteur réel ou virtuel), rigidement fixée à la caméra, qui délivre régulièrement des observations ${}^{\mathcal{W}_k}\mathbf{z}_i^k$ sur le déplacement de la caméra.

L'idée de la méthode est d'intégrer ces observations secondaires dans l'ajustement de faisceaux local de l'application de *Structure-from-Motion* séquentiel [Mouragnon 2006], en ajoutant un terme de pénalité (ou contrainte) pondéré sur chaque pose clés de la caméra. La contrainte est construite à partir des observations livrées par une seconde source et est généralement une distance (au carré) pondérée (par un facteur noté λ_i) entre les paramètres de la pose courante $\mathcal{T}_i$ de la caméra et les valeurs prédites par les observations du second capteur (la pose $\hat{\mathcal{T}}_i^k$, ou seulement une partie).

5.2.1.2 Exemple de contraintes et leur application

Nous donnons ici à titre indicatif quelques contraintes possibles sur les paramètres de la caméra (intrinsèques et extrinsèques). Notons que des contraintes sur la géométrie de la scène (plans, points 3D de référence, etc.) sont envisageables mais n'ont pas été étudiées dans cette thèse. Le tableau 5.1 répertorie quelque unes des contraintes les plus courantes dans différents contextes : contraintes absolues, de déplacement, de lissage ou d'autocalibrage.

Notons que la solution que nous proposons implique que la contrainte suive une hypothèse de distribution normale centrée $\varepsilon_i^k(\mathbf{x}) \in \mathcal{N}(\mathbf{0}, \sigma^2\mathcal{I})$, dont la variance σ^2 est indéterminée ou imprécise. En pratique cette erreur représente une mesure de distance au carré entre des prédictions

1. Ce terme d'erreur correspond au lissage de la trajectoire proposé par FARENZENA *et al.* [Farenzena 2008]
2. Ces termes d'erreur correspondent à un processus d'autocalibrage d'une caméra dont les paramètres intrinsèques varient légèrement entre les images (voir [Kahl 2001])

Type	Paramètres contraints	Obs. $\hat{\mathbf{z}}_i^k$	Contrainte $\varepsilon_i^k(\mathbf{x})$
Absolu	Pose	$\hat{\mathcal{T}}_i^k$	$\varepsilon_i^{k,P}(\mathbf{x}) = \|\mathcal{T}_i^{-1}\hat{\mathcal{T}}_i^k - \begin{pmatrix}\mathcal{I}_3 & \mathbf{0}\end{pmatrix}\|^2$
	Position	$\hat{\mathbf{t}}_i^k$	$\varepsilon_i^{k,t}(\mathbf{x}) = \|\mathbf{t}_i - \hat{\mathbf{t}}_i^k\|^2$
	Orientation *(matrice)*	$\hat{\mathcal{R}}_i^k$	$\varepsilon_i^{k,R}(\mathbf{x}) = \|\mathcal{R}_i^\top \hat{\mathcal{R}}_i^k - \mathcal{I}_3\|^2$
	Orientation *(reprojections)*	$\hat{\mathcal{P}}_i^k = \begin{pmatrix}\hat{\mathcal{R}}_i^k & \mathbf{t}_i\end{pmatrix}$	$\varepsilon_i^{k,\Psi_R}(\mathbf{x}) = \frac{1}{N_{\nu_i}} \sum_{j=1}^{N_n} \nu_j^i d^2(\mathcal{P}_i\mathbf{Q}_j, \hat{\mathcal{P}}_i^k\mathbf{Q}_j)$
Déplacement	Pose	$\hat{d\mathcal{T}}_i^k$	$\varepsilon_i^{k,dP}(\mathbf{x}) = \|(\mathcal{T}_{i-1}^{-1}\mathcal{T}_i)^{-1}\hat{d\mathcal{T}}_i^k - \begin{pmatrix}\mathcal{I}_3 & \mathbf{0}\end{pmatrix}\|^2$
	Translation	$\hat{\mathbf{dt}}_i^k$	$\varepsilon_i^{k,dt}(\mathbf{x}) = \|\mathbf{t}_i - \mathbf{t}_{i-1} - \hat{\mathbf{dt}}_i^k\|^2$
	Rotation *(matrice)*	$\hat{d\mathcal{R}}_i^k$	$\varepsilon_i^{k,dR}(\mathbf{x}) = \|(\mathcal{R}_{i-1}^\top \mathcal{R}_i)^\top \hat{d\mathcal{R}}_i^k - \mathcal{I}_3\|^2$
	Rotation *(Euler)*	$\hat{\mathbf{de}}_i^k$	$\varepsilon_i^{k,de}(\mathbf{x}) = \|\operatorname{rot}_{\mathrm{e}}^{-1}\left(\mathcal{R}_{i-1}^\top \mathcal{R}_i\right) - \hat{\mathbf{de}}_i^k\|^2$
	Échelle	$\hat{s}_i^k = \|\hat{\mathbf{t}}_i^k\|$	$\varepsilon_i^{k,s}(\mathbf{x}) = \left\|\hat{s}_i^k - \|\mathbf{t}_i - \mathbf{t}_{i-1}\|\right\|^2$
Lissage	Matrice de projection	$\hat{\mathcal{P}}_i^k = \mathcal{P}_{i-1}$	$\varepsilon_i^{k,l}(\mathbf{x}) = \|\mathcal{P}_i - \hat{\mathcal{P}}_i^k\|^2$
	Mat. Proj. *(reprojections)*[1]	$\hat{\mathcal{P}}_i^k = \mathcal{P}_{i-1}$	$\varepsilon_i^{k,\Psi_l}(\mathbf{x}) = \frac{1}{N_{\nu_i}} \sum_{j=1}^{N_n} \nu_j^i d^2(\mathcal{P}_i\mathbf{Q}_j, \hat{\mathcal{P}}_i^k\mathbf{Q}_j)$
Autocalibrage[2]	Focale *(relatif)*	$\hat{f}_i^k = f_{i-1}$	$\varepsilon_i^{k,f}(\mathbf{x}) = \log(\frac{\hat{f}_i^k}{f_i})^2$
	Point principal	$\hat{\mathbf{c}}^k$	$\varepsilon_i^{k,c}(\mathbf{x}) = \|\mathbf{c}_i - \hat{\mathbf{c}}^k\|^2$

TABLE 5.1 – Exemples de contraintes sur les paramètres intrinsèques et extrinsèques de la caméra i construites à partir des observations issues d'une seconde source ou d'*a priori*.

ou observations du second capteur et les paramètres optimisés par l'ajustement de faisceaux. Par exemple un choix possible de distance entre deux orientations est la différence entre la matrice de rotation entre ses deux orientations et la matrice identité (on évite ainsi les discontinuités apportées par une différence d'angles d'Euler. Notons de plus qu'il est préférable que la contrainte soit une erreur moyenne (MSE : *Mean Square Error*) afin que le nombre de paramètres contraints n'influence pas l'importance de la contrainte vis-à-vis du critère image.

Il existe différentes applications nécessitant l'intégration de contraintes sur les paramètres de la caméra. Par exemple les applications de fusion multi-capteur contraignent les paramètres extrinsèques de la caméra avec des informations provenant de capteurs absolus (GPS, magnétomètres), ou contraignent le déplacement entre deux caméras avec des capteurs proprioceptifs (IMU, odomètres), afin de limiter les dérives de localisation. Nous étudions ce cas dans la dernière section 5.4. Certains auteurs [Farenzena 2008] tentent de réduire les problèmes d'estimation dus aux mouvements critiques de la caméra en réalisant un lissage (*smoothing*) de la trajectoire, et utilisent une contrainte exprimée en pixels afin d'avoir la même unité que l'erreur de reprojection de l'ajustement de faisceaux.

Une autre application, l'autocalibrage, fait intervenir les paramètres intrinsèques de la caméra, où l'on désire estimer la focale, qui évolue lors d'un zoom. Le principe est alors de contraindre la variation de celle-ci au cours du temps (voir [Kahl 2001]).

Dans cette thèse nous avons testé cette méthode de fusion générique dans le cadre de la fusion multi-capteur (IMU, odomètres) pour la localisation temps réel (SLAM) afin de tenter de limiter les dérives liées à la localisation par vision monoculaire pure. Mais avant de présenter comment cette technique peut être utilisée dans ce contexte, nous décrivons comment nous estimons automatiquement les coefficients de pondérations.

5.2.2 Principes de l'approche proposée

Considérons un ajustement de faisceaux local dans lequel est associé à chaque pose de la caméra, une contrainte issue des informations provenant par exemple d'un second capteur, ou d'un algorithme complémentaire, symbolisé par la lettre k. Une vue illustrée de la configuration est données dans la figure 5.2.

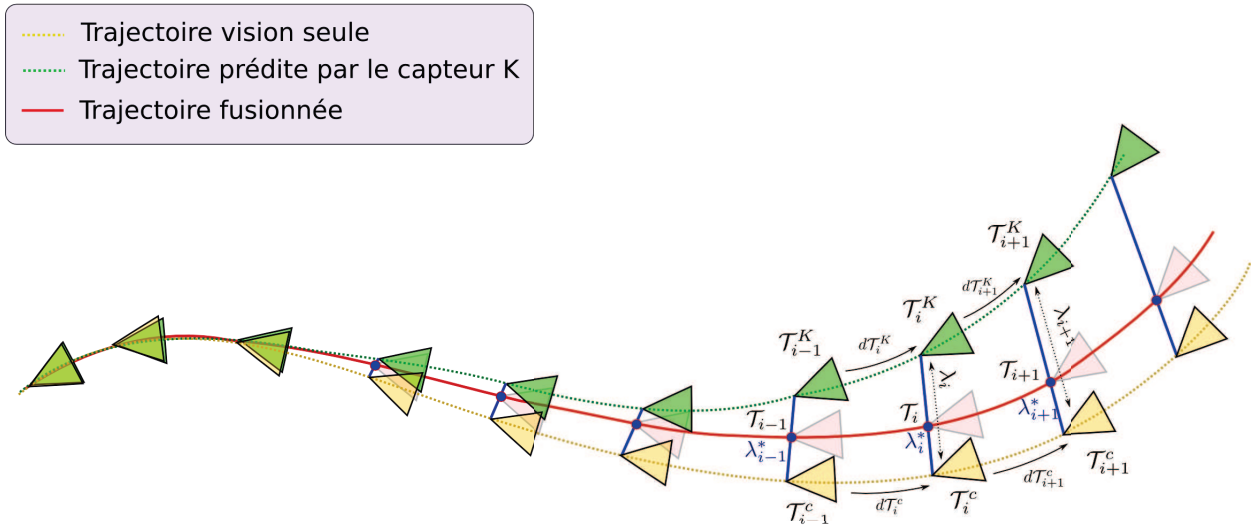


FIGURE 5.2 – Illustration de la fusion de la trajectoire estimée par un algorithme de *Structure-from-Motion* séquentiel avec les observations d'un second capteur k et par l'estimation dynamique des coefficients de pondération $\lambda_{i\in[1..N_m]}$.

5.2.2.1 Fonction de coût spécifique

L'approche généralement employée, est celle qui consiste à réunir en un seul et unique critère (l'*Aggregate Objective Function*), à la fois l'information image (caméra c) et l'information liée au deuxième capteur k. Cela est réalisé par une combinaison linéaire des deux objectifs : les erreurs de reprojection des points 3D vus dans la pose i : $\varepsilon^c_i(\mathbf{x}_i)$ et le terme de contrainte pondérée associé : $\varepsilon^k_i(\mathbf{x}_i)$. Il est nécessaire ici d'introduire un paramètre de pondération λ_i (image de l'isovariance de la contrainte) étant donné que les deux fonctions de coût ne partagent pas nécessairement la même unité (des pixels pour le critère de vision et des radians par exemple pour une contrainte sur les angles d'Euler), le même ordre de grandeur, ni la même confiance. La valeur du coefficient de pondération va indiquer à la minimisation d'accorder plus ou moins d'importance à la contrainte. Ce critère composé associé à chaque pose clé du SLAM visuel, noté $\varepsilon^{c+k}_i(\mathbf{x}_i;\lambda_i)$, s'écrit :

$$\varepsilon^{c+k}_i(\mathbf{x}_i;\lambda_i) = \bar{\varepsilon}^c_i(\mathbf{x}_i) + \lambda_i^2 \varepsilon^k_i(\mathbf{x}_i), \tag{5.1}$$

où $\bar{\varepsilon}_i^c(\mathbf{x}_i)$ est le critère de vision, et plus particulièrement l'erreur moyenne (MSE) de reprojection des points 3D du vecteur $\mathbf{x}_i$:

$$\bar{\varepsilon}_i^c(\mathbf{x}_i) = \frac{1}{Card(j|\mathbf{Q}_j \in \mathbf{s}_i)} \sum_{j|\mathbf{Q}_j \in \mathbf{s}_i} \nu_j^i d^2(\hat{\mathbf{q}}_j^i, \mathcal{P}_i \mathbf{Q}_j), \tag{5.2}$$

et $\varepsilon_i^k(\mathbf{x}_i)$ est le second critère, la contrainte générique associée. Le paramètre $\lambda_i \in \mathbb{R}^+$ paramétrise la combinaison linéaire entre ces deux objectifs et permet de privilégier l'un ou l'autre des objectifs.

5.2.2.2 Ajustement de faisceaux local contraint

L'ajustement de faisceaux devient alors un problème multiobjectif formé des termes d'erreur de reprojection et des contraintes sur les paramètres des caméras clés, dans lequel nous souhaitons toujours estimer les poses clés de la caméra et la structure de la scène qui minimisent ces critères, mais nous devons aussi estimer de nouveaux paramètres inconnus, les coefficients de pondération des contraintes. La fonction de coût de l'ajustement de faisceaux local contraint est la suivante :

$$F_w^{LBA}(\mathbf{x}_i; \{\lambda_1, \ldots, \lambda_i\}) = \sum_{i'=1}^{N_m} \varepsilon_{i'}^{c+k}(\mathbf{x}_i; \lambda_{i'}). \tag{5.3}$$

À l'inverse des travaux de recherche précédents, nous considérons chaque contrainte de manière indépendante, c'est-à-dire que chaque contrainte possédera un coefficient de pondération qui lui est propre. Cela permet ainsi de gérer les cas où les incertitudes des observations fournies par le second capteur varient de manière importante au cours du temps. Une illustration de la configuration (trajectoire de la caméra estimée par vision et prédite par la seconde source) est donnée dans la figure 5.2.

5.2.2.3 Résolution séquentielle

Puisque nous nous plaçons dans le cadre d'un algorithme de *Structure-from-Motion* séquentiel temps réel dans lequel les images sont traitées incrémentalement, les pondérations de chaque nouvelle pose clé de la caméra doivent être estimés dynamiquement, dès que la nouvelle pose est estimée par les algorithmes de vision. Nous décomposons ainsi le problème multiobjectif (plusieurs pondérations inconnues) en plusieurs ajustement de faisceaux bi-objectif dans lesquels seul le dernier poids est à déterminer.

La résolution des problèmes bi-objectifs (composés de deux objectifs ; erreurs de reprojections et contrainte associée) est un problème difficile (voir la section 2.2.3), et est habituellement traité en deux étapes consécutives : estimation d'un poids efficace pour la contrainte d'une part, puis optimisation classique avec la combinaison des termes d'erreur pondérés. L'algorithme 10 (`SfMIncBiobjectifBA`) montre le principe de localisation incrémentale avec contraintes basée sur l'ajustement de faisceaux contraint.

Algorithme 10 : `SfMIncBiobjectifBA` : Exemple de *Structure-from-Motion* incrémental avec un ajustement de faisceaux bi-objectif

- Entrées : Une séquence d'images et les observations extérieures associées
- Sorties : Une trajectoire (poses d'une caméra) et une scène (ensemble de points 3D)

pour *chaque nouvelle image* i **faire**

`// Resection (Structure-from-Motion)`
Estimer la pose $\mathcal{T}_i^c$ (et $\mathcal{P}_i^c$) par vision (section 3.1.4) ;
`// Contrainte externe`
Récupérer la contrainte associée $\varepsilon_i^{k_i}(\mathbf{x})$ (section 5.3.4);
`// Sélection de la pondération`
Estimer le coefficient de pondération λ_i associé à la contrainte, à partir des données précédentes (section 5.2.3) ;
`// Construction de la scène (triangulation)`
Trianguler les nouveaux points 3D de la scène (section 3.1.2);
`// Optimisation finale`
Effectuer l'ajustement de faisceaux local (5.12) avec les contraintes pondérées associées (section 5.2.4) ;

Les coefficients de pondération sont estimés incrémentalement, à l'apparition de la pose clé. Lorsque la nouvelle pose i est détectée, le poids λ_i de la contrainte associée ε_i^k est estimé à l'aide d'une des différentes techniques de sélection de modèle, par apprentissage ou par critères de compromis (voir la section 5.2.3). Une illustration du principe de la sélection du poids d'une contrainte sur 2 itérations ($i-1$ et i) est présentée dans la figure 5.3.

Une fois que le poids de la nouvelle pose est estimé par une des techniques proposées dans la section 5.2.3, un ajustement de faisceaux local est ensuite effectué sur les paramètres les plus récents (trajectoire et scène) avec la fonction de coût composée des erreurs de reprojection de chaque pose de la caméra auxquelles nous ajoutons les contraintes pondérées associées.

Notons que le tout premier ajustement de faisceaux est effectué après la reconstruction initiale (voir 3.3.2), le premier triplet et la première reconstruction de la scène à la troisième itération : $\mathbf{x}_{i=3} = [\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3\mathbf{s}_3]$, et dont la fonction de coût est seulement basée sur l'image : $F_w^{LBA}(\mathbf{x}_3) = F^c(\mathbf{x})$ (de l'équation (3.26)).

Les sections suivantes (5.2.4 et 5.2.3) décrivent en détail les différents processus mis en œuvre.

5.2.3 Sélection pratique des coefficients de pondération

La principale difficulté liée à l'intégration de contraintes dans l'ajustement de faisceaux est le choix des coefficients de pondération associés $\lambda_{i\in[1,N_m]}$. Nous l'avons vu, la plupart du temps ces coefficients sont choisis pour être égaux et leur valeur est estimée empiriquement, par tâtonnement. Cette méthode manque clairement d'adaptation à la configuration souvent dynamique

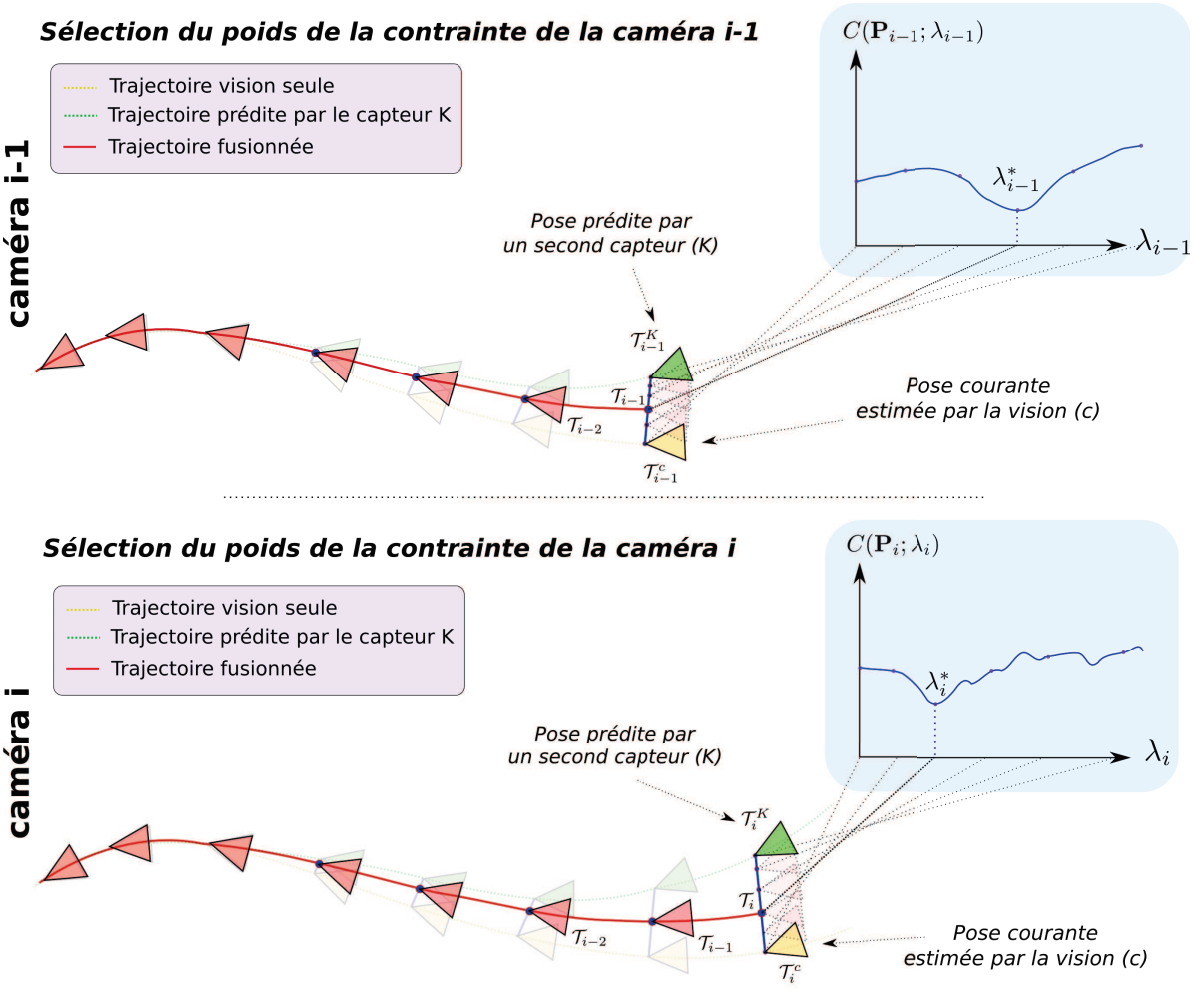


FIGURE 5.3 – Illustration de l'estimation incrémentale des coefficients de pondération λ_{i-1} et λ_i. Ces valeurs correspondent aux maxima des scores estimés pour plusieurs valeurs de λ.

du problème, et nous pensons qu'une estimation dynamique et indépendante des poids des contraintes est bénéfique à la qualité de la localisation.

5.2.3.1 Principe de la méthode

L'estimation des coefficients de pondération est réalisée incrémentalement. Lorsqu'une nouvelle pose $\mathcal{T}_i$ est ajoutée à l'ajustement de faisceaux, nous estimons le poids λ_i de la contrainte associée $\varepsilon^k_i(\mathbf{x})$.

Il est impossible d'estimer simultanément les paramètres $\mathbf{x}_i$ et le coefficient de pondération λ_i : $\min_{\mathbf{x}_i,\lambda_i} F^{LBA}_w(\mathbf{x}_i;\{\lambda_1,\ldots,\lambda_i\})$. En effet, il est évident que cette minimisation tentera d'annuler la pondération ($\lambda^*_i = 0$) afin d'éliminer le coût supplémentaire apporté par la contrainte. Il faut donc utiliser d'autres techniques pour sélectionner le poids de la contrainte. L'approche classique pour résoudre ce problème imbriqué (équation (5.4)) est composée de deux étapes :

la sélection du meilleur poids minimisant un critère de qualité C, puis l'optimisation de la pose avec la fonction objectif composée. Le critère $C(\mathbf{x}_i; \lambda_i)$ évalue la qualité des paramètres $\mathbf{x}_i$ du modèle étant donné le paramètre de pondération λ_i. La recherche du meilleur poids λ_i^* doit dans ces conditions être effectuée en discrétisant l'espace de recherche du coefficient de pondération, et estimer ponctuellement le critère pour chaque point de l'espace. Ainsi, le nombre d'appels au critère peut être important et peut nécessiter un temps de calcul important pour estimer l'ensemble des paramètres courants $\mathbf{x}_i$. Puisque nous ciblons une application temps réel, la sélection du paramètre de pondération doit être rapide (quelques centaines de millisecondes au maximum) et nous proposons de diminuer le nombre de variables optimisées simultanément, en se limitant aux paramètres de la pose courante. Le critère $C(\mathbf{P}_i; \lambda_i)$ se trouve être bien plus rapide que le critère $C(\mathbf{x}_i; \lambda_i)$ basé sur l'ensemble du vecteur $\mathbf{x}_i$.

Ajustement bi-objectif de la pose. Nous proposons une heuristique permettant de s'affranchir de sélectionner le coefficient de pondération en optimisant l'ensemble des paramètres de l'ajustement de faisceaux local. La sélection du poids ne s'effectue pas dans un ajustement de faisceaux local bi-objectif, mais seulement lors de l'optimisation de la seule dernière pose, paramétrée par le vecteur $\mathbf{P}_i$ à partir des points 3D de la scène qui ont déjà été optimisés lors des précédentes itérations. L'estimation conjointe de la pose $\mathbf{P}_i$ et le poids λ_i s'effectue en résolvant le problème imbriqué suivant :

$$\min_{\mathbf{P}_i} \varepsilon_i^{c+k} \left(\mathbf{x}_i; \arg\min_{\lambda_i \geq 0} C(\mathbf{P}_i; \lambda_i) \right). \tag{5.4}$$

L'estimation du coefficient de pondération est réalisée sur cette fonction de coût restreinte, ne faisant intervenir que les erreurs de reprojection de l'image courante sur les anciens points 3D et non l'ensemble des images, ainsi que le terme de contrainte associé à la pose.

La première étape consiste à résoudre le sous problème de l'équation (5.4) :

$$\lambda_i^* = \arg\min_{\lambda_i \geq 0} C(\mathbf{P}_i; \lambda_i). \tag{5.5}$$

En pratique, nous avons généré une séquence logarithmique de $N_\lambda - 1$ valeurs positives sur le domaine $0 < \lambda^{min} \geq \lambda^{max}$, plus la valeur $\lambda = 0$. L'ensemble des hypothèses de poids générées est le suivant :

$$L = \{0, 10^{\log(\lambda^{min}) + \frac{k-1}{N_\lambda}(\log(\lambda^{max}) - \log(\lambda^{min}))}\}_{k \in [1, N_\lambda - 1]}. \tag{5.6}$$

Les paramètres de λ^{min} et λ^{max} sont choisis empiriquement à partir des expérimentations et dépendent du type des contraintes ajoutées dans la fonction de coût, et notamment de l'ordre de grandeur et de l'échelle des deux termes d'erreur. Il est important de choisir des valeurs adaptées afin de ne pas parcourir inutilement les zones extrêmes de l'espace. Il est préférable d'augmenter la résolution du pas de discrétisation de l'espace afin d'améliorer l'estimation du meilleur poids. En général, à partir d'une certaine valeur maximale, le terme d'erreur de la contrainte est très faible et il n'est pas utile de dépasser celle-ci (voir la figure 5.6(b)). Les valeurs $\lambda^{min} = 10^{-3}$ et $\lambda^{max} = 10^3$ semblent expérimentalement suffisantes pour la majorité des contraintes utilisées (sur l'échelle $\varepsilon_i^{k,s}$ et sur les orientations $\varepsilon_i^{k,dR}$).

Le choix du nombre de coefficients de pondération N_λ ou, de façon équivalente, du pas de la discrétisation, dépend du compromis entre la qualité de l'estimation désirée et le temps accordé à la méthode de sélection et de la complexité de calcul du critère. Puisque nous destinons cette approche au SLAM temps réel, la sélection du poids doit être rapide et nous proposons de considérer et évaluer $N_\lambda = 100$ valeurs de pondération.

La résolution du problème d'optimisation de la pose lorsque le poids de la contrainte est identifié est réalisée par la méthode de LEVENBERG-MARQUARDT de la manière suivante :

$$\mathbf{P}_i^* = \arg\min_{\mathbf{P}_i} \varepsilon_i^{c+k}(\mathbf{x}_i; \lambda_i), \tag{5.7}$$

avec la solution initiale $\mathbf{P}_i^{(0)} = \mathbf{P}_i^c$.

5.2.3.2 Différents critères de sélection

Nous avons comparé plusieurs critères génériques de sélection de modèle ou de complexité, dans le cadre de la sélection du coefficient de pondération. Deux principales approches ont été étudiées : la sélection par apprentissage et par critère de compromis.

Apprentissage par validation croisée. La première approche proposée est l'approche par apprentissage. L'idée sous-jacente de cette méthode est que les paramètres du modèle optimisés, la pose $\mathbf{P}_i$, doivent pouvoir expliquer l'ensemble des observations $\hat{\mathbf{z}}_i$, lorsqu'elle est estimée à partir d'un sous-échantillon de $\hat{\mathbf{z}}_i$. Il s'agit ici typiquement d'un problème d'apprentissage supervisé, dans lequel la qualité du coefficient de pondération est définie en fonction de son aptitude à prédire les observations manquantes. L'approche proposée, par validation croisée (2.4.1), consiste à décomposer l'ensemble des observations $\hat{\mathbf{z}}_i^c$ de la dernière image i en deux sous-ensembles d'apprentissage et de test, puis de sélectionner le poids λ_i qui minimise l'erreur réalisée sur le jeu de test.

Cette technique est très employée de nos jours dans différents contextes [Bishop 2007]. Dans notre application, le vecteur d'observations $\hat{\mathbf{z}}_i$ est composé principalement des points d'intérêt $\hat{\mathbf{z}}_i^c = \{\hat{\mathbf{q}}_j^i\}$ du capteur caméra c, auxquels nous ajoutons les observations (synchronisées) du second capteur k : $\hat{\mathbf{z}}_i^k$ (voir le tableau 5.1). Or, ces dernières sont trop peu nombreuses pour que l'on puisse les utiliser dans l'apprentissage, les information redondantes proviennent seulement de la caméra. Ce choix de l'apprentissage seulement sur les données images va déterminer l'influence que va avoir la contrainte pondérée. En effet, dans ce contexte, la méthode de validation croisée va choisir le paramètre de pondération qui maximisera la prédictivité du modèle (la pose $\mathbf{P}_i$) sur la vision seulement. Les deux objectifs ne sont pas traités de manière identique, le principe ici est de voir comment "réagit" la partie vision (sa prédictivité) lorsque l'on ajoute la contrainte.

Étant donné le faible nombre d'observations images ($\sim$ 40 *inliers* par pose parmi 200-300 points 2D) nous a encouragé à opter pour la variante *leave-one-out* de la validation croisée. Cette version recycle le jeu de test, constitué d'une seule observation, dans les apprentissage suivants, ce qui a pour effet de moyenner l'erreur de test sur l'ensemble des observations.

Le critère de la validation *leave-one-out* $C_{CV_{loo}}(\mathbf{P}_i;\lambda_i)$ reflète l'erreur de prédiction pour le poids λ_i et représente la différence moyenne entre l'observation image $\hat{\mathbf{q}}_j^i$ et sa valeur prédite $\mathbf{q}_j^i$, c'est-à-dire la projection de point 3D $\mathbf{Q}_j$ dans l'image i : $\mathbf{q}_j^i = \Psi(\mathcal{P}_i^{[j]}\mathbf{Q}_j)$, où $\mathcal{P}_i^{\{j\}}$ est estimé sans cette observation, en résolvant :

$$\mathbf{P}_i^{\{j\}} = \arg\min_{\mathbf{P}_i^{[j]}} \varepsilon_i^{c+k}([\mathbf{P}_i\mathbf{s}_i^{\{j\}}];\lambda_i), \tag{5.8}$$

où $\mathbf{s}_i^{\{j\}}$ est l'ensemble des points 3D, sans le point $\mathbf{Q}_j$. L'équation (5.8) est résolue par plusieurs itérations de GAUSS-NEWTON ou LEVENBERG-MARQUARDT. En pratique, étant donné le grand nombre d'appels à cette minimisation ($N_{\nu_i}N_\lambda$ fois !), nous avons opté pour un nombre limité (3) d'itérations de LEVENBERG-MARQUARDT.

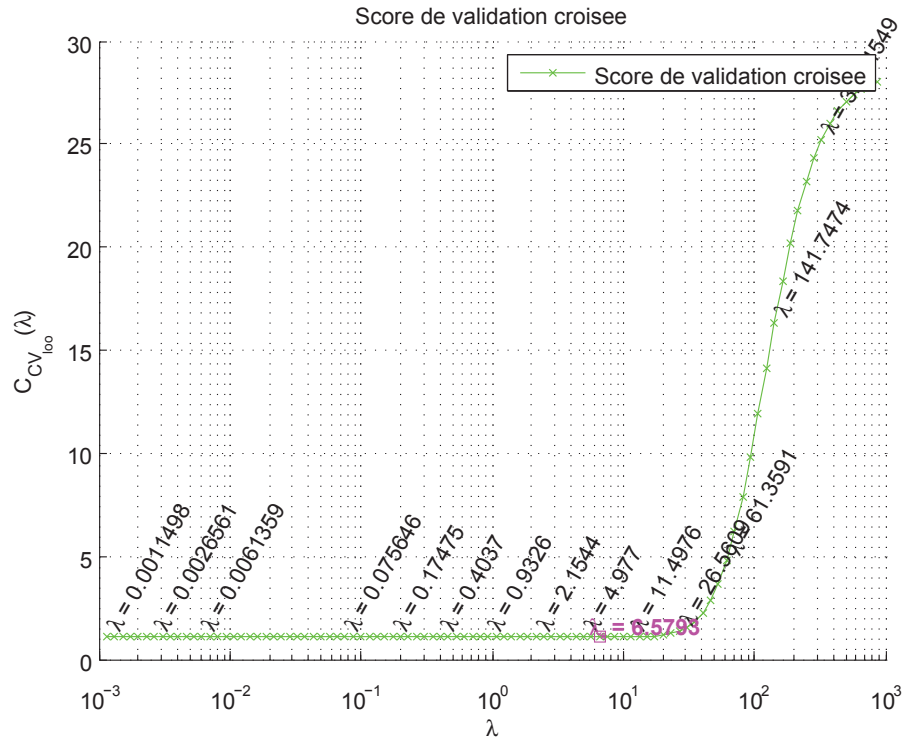
Le critère de la variante *Leave-one-out* de la validation croisée est donc la moyenne des erreurs de prédiction :

$$C_{CV_{loo}}(\mathbf{P}_i;\lambda_i) = \frac{1}{|(\forall j|\nu_j^i = 1)|}\sum_{j=1}^{N_n}\nu_j^i d^2(\hat{\mathbf{q}}_j^i, \mathcal{P}_i^{\{j\}}\mathbf{Q}_j). \tag{5.9}$$

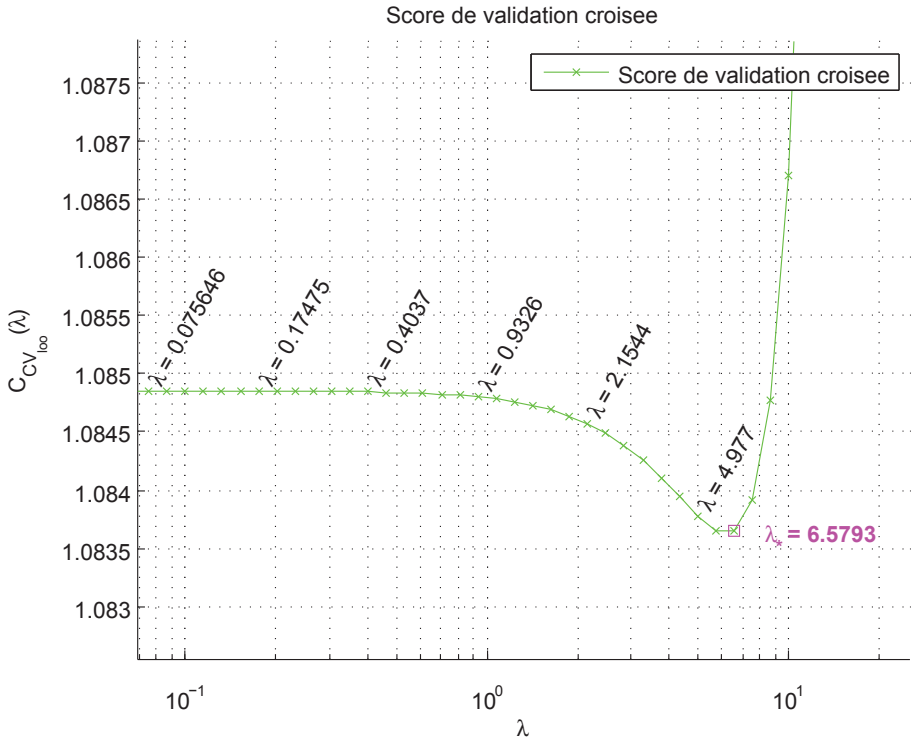
L'algorithme 2 page 58 présentant le calcul de ce critère. La figure 5.4 représente la critère $C_{CV_{loo}}(\mathbf{P}_i;\lambda_i)$ en fonction de la pondération λ_i, pour une contrainte d'orientation ε_R^k dans le contexte d'un SLAM multi-capteur (voir la section 5.3) et à la septième image clé ($i = 7$). L'ensemble des pondérations estimées sur la séquence est présenté dans la figure 5.8.

Nous pouvons remarquer qu'il existe une valeur de pondération pour laquelle la prédictivité de la pose $\mathbf{P}_7$ est améliorée, il s'agit de la valeur $\lambda_7^* \approx 6.58$. Cette amélioration intervient juste avant l'importante dégradation de la prédictivité que l'on peut apercevoir dans la figure 5.4(a).

Ce critère de sélection du poids a l'avantage d'être particulièrement stable par rapport aux critères de compromis puisqu'il ne fait pas intervenir de dérivé. Comme nous employons ici la variante *leave-one-out* de la validation croisée, dont le jeu de test n'est constitué que d'un seul point 3D (recyclé), la variation du score entre $\lambda_i = 0$ et $\lambda_7^* \approx 6.58$ est très faible. Elle est de l'ordre de 0.05px, ce qui est bien inférieur à ce que l'on obtient avec une forte pondération (30px). Néanmoins, la discrétisation de l'espace du coefficient de pondération est assez fine pour que l'on puisse détecter ce point particulier.



(a) Critère de la validation croisée *leave-one-out*.

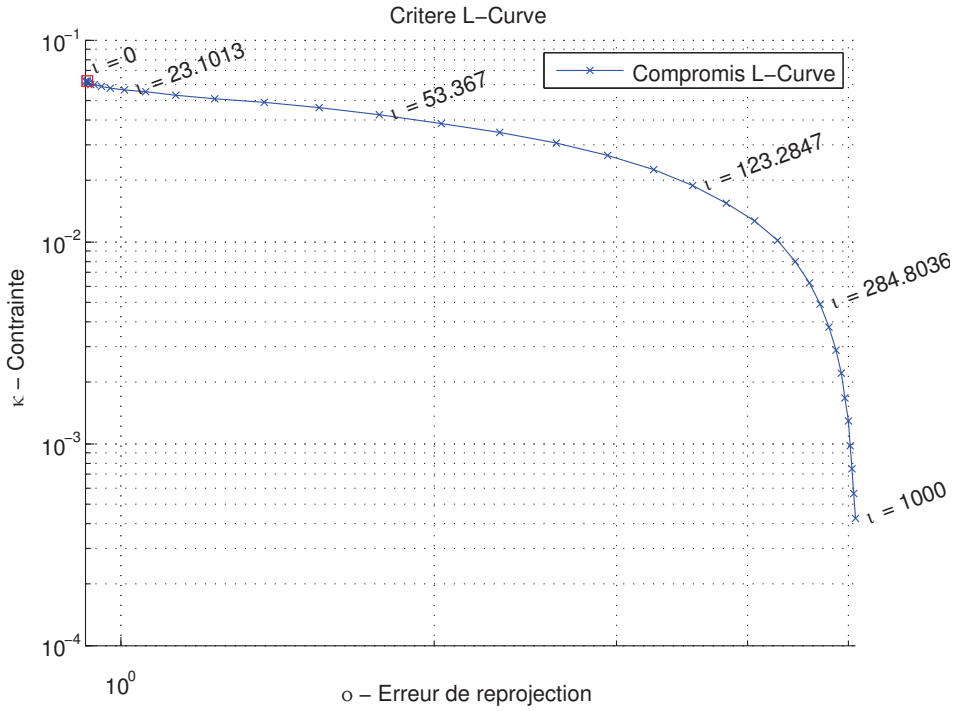


(b) Critère de la validation croisée *leave-one-out* (zoom).

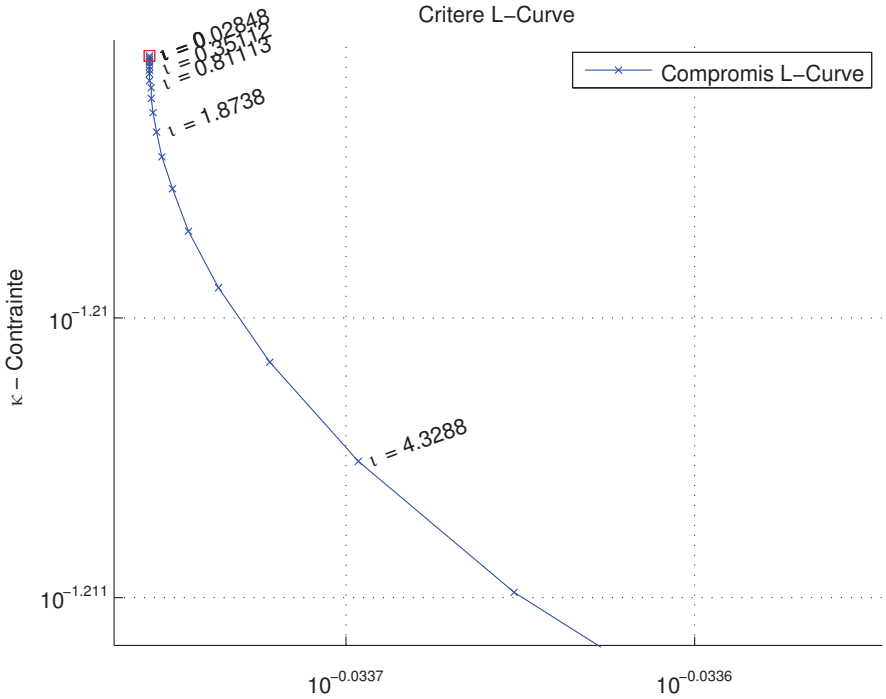
FIGURE 5.4 – Exemple de critère par validation croisée *leave-one-out* (a) (et zoom (b)) avec un terme de contrainte d'orientation issue d'une centrale de navigation (INS), à la septième itération.

Critères de compromis. Nous avons aussi étudié sur ce problème d'estimation de coefficient de pondération deux critères originellement issus de la régularisation de problèmes mal posés : les critères de compromis (voir la section 2.4.2 page 59). À la différence de la méthode de validation croisée dont le critère est construit sur la qualité de prédiction de nouvelles données apportée par la pose optimisée avec la contrainte pondérée, les critères de compromis tentent d'estimer le meilleur compromis (*trade-off*) entre les deux objectifs $\rho = \varepsilon_i^c(\mathbf{x}_i)$ (erreurs de reprojection de la pose $\mathbf{P}_i$) et $\nu = \varepsilon_i^k(\mathbf{x}_i)$ (la contrainte associée). L'idée est de rechercher la pondération la plus impartiale, ne favorisant aucun des deux objectifs. Un exemple de courbe de compromis sur un problème de SLAM visuel contraint par une centrale de navigation (INS) est donné dans la figure 5.5. La courbe correspond à l'estimation du poids de la contrainte associée à la pose $\mathbf{P}_7$.

Nous pouvons remarquer sur la figure 5.5(a) que, contrairement aux problèmes de régularisation classiques, la courbe de compromis n'est pas convexe sur l'ensemble de l'espace de recherche. La courbe est convexe seulement dans la première partie de l'espace, dans l'intervalle approximatif $[10^{-3}, 10]$. Au delà, la courbe change de courbure et devient concave. Ces deux intervalles délimitent les deux comportements distincts qu'adopte la courbe de compromis. Même si dans cet exemple la contrainte pondérée ne diminue jamais l'erreur de reprojection quelque soit la valeur de pondération, dans la première partie (voir la figure 5.5(b)), avec l'accroissent du poids, le terme de contrainte diminue bien plus vite que l'erreur de reprojection n'augmente. Ensuite, dans le second intervalle, le comportement est inversé : l'erreur de reprojection augmente bien plus vite que ne décroit le terme de contrainte. Notons que ce deuxième intervalle est aussi détecté par le critère de validation croisée (voir la figure 5.4(a)). La séparation entre les deux comportements n'est pas comme dans l'exemple de régularisation de Tikhonov (figure 2.6 page 60), bien nette et est difficilement observable. Tout l'enjeu ici est de trouver le poids à partir duquel le comportement s'inverse.



(a) Courbe de compromis.



(b) Courbe de compromis (zoom).

FIGURE 5.5 – Exemple de courbe de compromis en échelle logarithmique (a) (et zoom (b)) avec un terme de contrainte d'orientation issue d'une centrale de navigation, pour l'estimation du poids λ_7.

Pour estimer le meilleur poids séparant ces deux comportements, nous avons employé les critères L-Curve et LTN. Ces deux critères sont décrits dans la section 2.4.2 et sont définis par l'algorithme `CriterionBasedModelSelection`.

L'objectif de la méthode de L-Curve est de rechercher le point de séparation entre ces deux comportements qui maximise la courbure $\kappa(\lambda_i)$ de la courbe de compromis. Le critère de L-Curve est basé sur cette information de courbure : $C_{L-Curve}(\mathbf{P}_i;\lambda_i) = -\kappa(\lambda_i)$, où $\kappa(\lambda_i)$ est défini dans l'équation (2.53), avec $\hat{\rho} = \log(\varepsilon_i^c(\mathbf{x}_i))$ et $\hat{\nu} = \log(\varepsilon_i^k(\mathbf{x}_i))$. La figure 5.6(a) montre les évolutions des deux objectifs par rapport aux différentes hypothèses de pondération, et la figure 5.6(b) présente le critère de L-Curve, toujours sur l'exemple de la septième caméra clé.

Comme nous l'avons vu avec la figure 5.5(a), la séparation entre les deux comportements n'est pas nette (pas de parties verticales et horizontales), et la courbure du compromis présente souvent un pic de courbure assez effacé. Dans l'application considérée, le point de séparation correspondant au pic de courbure et au minimum du critère L-Curve est détecté pour de faibles valeurs de pondérations (voir la figure 5.6(b)). En pratique, cette zone est assez chaotique à cause des instabilités numériques dues aux dérivées secondes nécessaires à l'estimation de la courbure, à la discrétisation de l'espace et à la qualité des optimisations de LEVENBERG-MARQUARDT. Pour diminuer l'influence de ces instabilités et privilégier la détection d'une zone minimale plutôt que d'une valeur instable, nous utilisons la médiane locale des valeurs de courbure sur une fenêtre de $N = 11$ valeurs ($\text{mediane}_{l' \in [-N/2, N/2]}(\mathbf{x})$), telle que :

$$\forall l \in [N/2, N_\lambda - N/2], \quad \bar{C}_{L-Curve}(\mathbf{P}_i;\lambda_i^l) = \underset{l' \in [-N/2, N/2]}{\text{mediane}} (-\kappa(\lambda_i^{l+l'})). \tag{5.10}$$

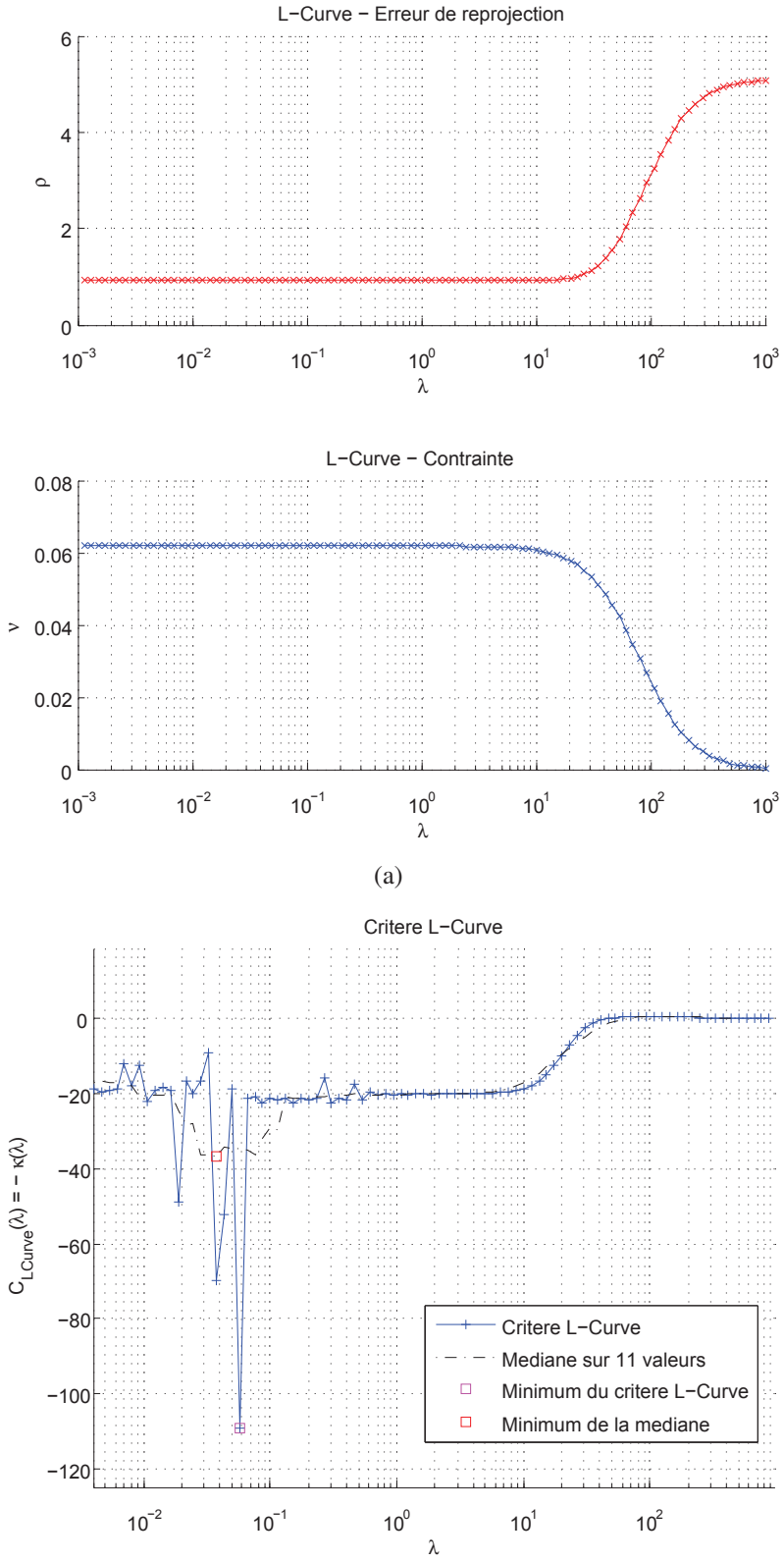


FIGURE 5.6 – Exemple de critère par L-Curve (a) avec un terme de contrainte d'orientation issue d'une centrale de navigation. La figure (a) représente l'évolution des deux objectifs (erreurs de reprojection et contrainte) par rapport aux hypothèses de poids et la figure (b), l'évolution du critère.

Le critère L-Curve étant particulièrement sensible aux instabilités numériques, nous avons choisi de tester une variante plus simple et ne faisant pas intervenir les dérivées secondes : le critère LTN (décrit en section 2.4.2.2).

L'objectif du critère LTN est de rechercher le poids doté de la plus faible norme de la tangente de la courbe de compromis, c'est-à-dire le point ayant la plus faible vitesse de décroissance. Les résultats de ce critère sur les mêmes données que les deux précédents critères (le poids de la pose $\mathbf{P}_7$) sont présentés dans la figure 5.7.

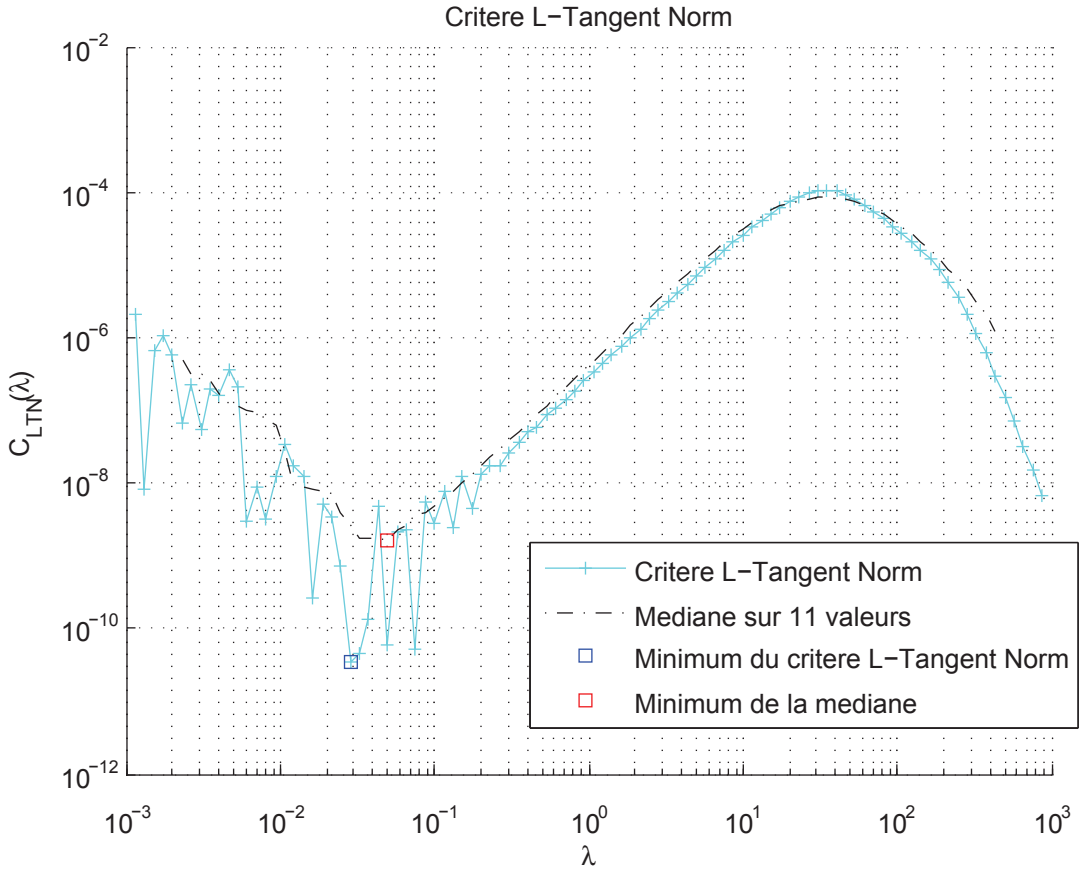


FIGURE 5.7 – Sélection du poids λ_7 par le critère LTN, avec un terme de contrainte d'orientation issue d'une centrale de navigation.

Nous pouvons remarquer que le critère LTN adopte un comportement similaire au critère L-Curve et présente une zone minimale approximativement identique ($[10^{-2}, 10^{-1}]$). Même si des instabilités sont présentes pour de faibles valeurs de pondération, celles-ci sont moins importantes qu'avec le critère L-Curve. Pour diminuer ces instabilités, nous réalisons un lissage du critère, sur une fenêtre de 11 valeurs de poids :

$$\forall l \in [N/2, N_\lambda - N/2], \quad \bar{C}_{LTN}(\mathbf{P}_i; \lambda_i^l) = \underset{l' \in [-N/2, N/2]}{\text{mediane}} (C_{LTN}(\mathbf{P}_i; \lambda_i^{l+l'})). \tag{5.11}$$

En revanche, le critère LTN est plus sensible que le critère L-Curve au fait que la courbe

de compromis soit concave pour des valeurs importantes de poids (le critère diminue dans cette partie de l'espace).

Vérification de la stabilité. Les critères de sélection de poids sont des heuristiques et il n'y a donc aucune garantie sur le fait que la valeur de pondération sélectionnée ne dégrade pas le processus de localisation incrémentale. Pour éviter la situation dans laquelle le choix de la pondération affaiblit la localisation incrémentale du SLAM visuel, nous procédons à une vérification sur le nombre de correspondances cohérentes (*inliers*).

Pour qu'une hypothèse de pondération soit acceptable, il faut que le nombre de points 3D pour lesquels l'erreur de reprojection après l'optimisation contrainte de la pose est inférieure à 2 pixels, soit supérieur à 90% du nombre d'inliers initial. Ces paramètres ont été estimés par essai-erreur sur plusieurs séquences. En pratique, cette méthode élimine les pondérations trop fortes qui entraînent une dégradation voire une perte totale de localisation due à une diminution des points 3D fiables (*inliers*) du SLAM visuel.

Comparaison des méthodes de sélection. Comme nous l'avons précisé, les résultats de ces trois techniques de sélection de poids proviennent de l'intégration d'informations d'orientation provenant d'un système de navigation à guidage inertiel, dans le contexte d'un SLAM multi-capteur (séquence `Gyroviz`, voir la section 5.3). Plus précisément il s'agit de l'estimation d'un poids choisi au hasard, le poids λ_7.

Estimation du poids. Les résultats précédents sur la sélection du poids pour les trois méthodes ont montré que les deux types de sélection de poids par apprentissage et par compromis sélectionnaient des poids assez différents (6.78 pour la validation croisée *leave-one-out* et environ 0.05 pour les critères basés sur le compromis). La différence est importante (de l'ordre de 10^2) et se retrouve sur l'ensemble des contraintes sur la séquence. La figure 5.8 présente les valeurs de poids estimées par les trois méthodes pour chaque image clé. Nous présentons ici seulement l'estimation des poids, sans que la contrainte ne soit réellement ajoutée une fois le poids sélectionné. Cela permet de comparer les méthodes sur des données et solutions initiales identiques. La ligne verticale rouge symbolise la septième image clé, correspondant aux figures précédentes (5.4, 5.5 et 5.6). Notons que les valeurs de poids nulles ne sont pas affichées sur ce graphique en échelle logarithmique, et correspondent aux discontinuités des scores.

Le tableau 5.2 donne les statistiques de la sélection des poids sur toutes les images clés, pour chaque méthode de sélection.

Sur ce graphique, nous observons d'une part que les méthodes L-Curve et LTN estiment des poids approximativement identiques, variant autour de 10^{-2}, ce qui est confirmé par les statistiques. La validation croisée *leave-one-out* calcule des poids bien différents, plutôt aux alentours de 5, avec de fortes variations. Bien que les deux types de critère détectent, par un accroissement de leur score C, la même zone dans laquelle les poids détériorent l'optimisation de la pose, elles ne sélectionnent pas le même point, optimal selon leur propre critère.

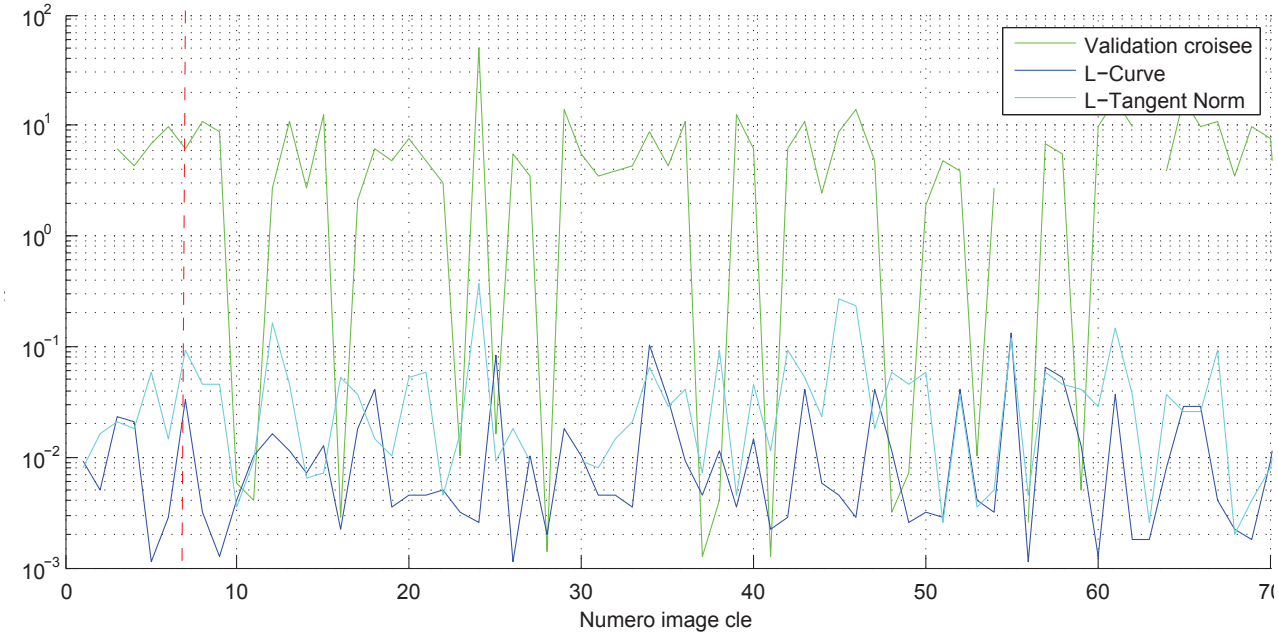


FIGURE 5.8 – Évolution des coefficients de pondération sélectionnés par les 3 méthodes (de validation croisée *leave-one-out*, L-Curve et LTN), pour des contraintes de rotation inter-caméras clés ($\varepsilon_i^{k,dR}$). La ligne rouge symbolise la 7ième image clé.

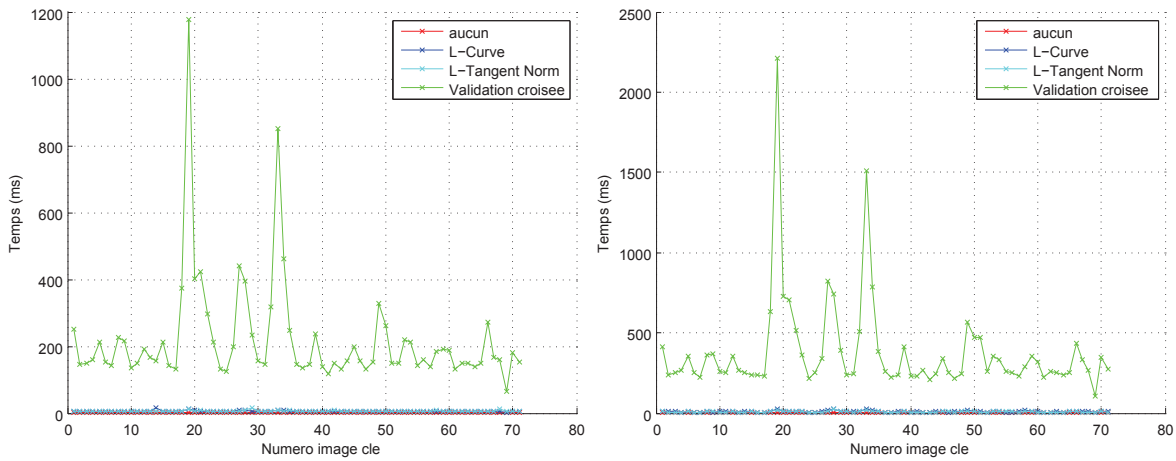
Méthode de sélection	**Médiane** *(sans unité)*	**Moyenne** *(sans unité)*	**Écart-type** *(sans unité)*
L-Curve	0.0061	0.015	0.023
LTN	0.0163	0.045	0.063
Validation croisée	4.9770	5.947	6.931

TABLE 5.2 – Valeurs moyennes et écarts types des poids estimés par les trois méthodes de sélection (Validation croisée, L-Curve, LTN) sur la séquence `Gyroviz` 5.4.3.1. Ces valeurs sont estimées sur une contrainte d'orientation ($\varepsilon_i^{k,dR}(\mathbf{x})$) parmi 100 hypothèses allant de $\lambda^{min} = 10^{-3}$ à $\lambda^{max} = 10^3$.

Temps de calcul. Les méthodes de sélection par validation croisée et par critère de compromis ont une complexité différente. En effet, alors que les critères de compromis ont besoin d'effectuer une seule optimisation de la pose par hypothèse de poids (N_λ en tout), la validation croisée *leave-one-out* doit en réaliser beaucoup plus : N_{ν_i} fois plus (soit $N_{\nu_i} N_\lambda$ en tout) !

La figure 5.9 et le tableau 5.3 présentent les temps de calcul utilisés par chacune des méthodes de sélection pour des contraintes d'orientation basées sur les matrices ($\varepsilon_i^{k,dR}(\mathbf{x})$) ou sur les reprojections ($\varepsilon_i^{k,\Psi_{dR}}(\mathbf{x})$). Les résultats de la méthode *aucune* représentent les temps de calcul d'une seule optimisation de la pose de la caméra, pour comparaison.

Ces résultats montrent que les méthodes de sélection de poids par critère de compromis ne consomment que peu de temps de calcul et peuvent être directement intégrées à un algorithme de SLAM temps réel, d'autant plus que ce temps de calcul n'est nécessaire que lorsqu'une nouvelle image clé est détectée.



(a) (b)

FIGURE 5.9 – Temps de calcul de la sélection du coefficient de pondération parmi 100 valeurs, avec une contrainte d'orientation (en matrices 5.9(a) et en reprojection 5.9(b)), pour les méthodes de validation croisée *leave-one-out*, L-Curve et LTN. Les courbes rouges représentent les temps d'optimisation sans la sélection de poids.

	Contrainte d'orientation			
	$\varepsilon_i^{k,R}(\mathbf{x})$		$\varepsilon_i^{k,\Psi_R}(\mathbf{x})$	
Méthode de sélection	Moyenne (ms)	Écart-type (ms)	Moyenne (ms)	Écart-type (ms)
aucune	0.103	0.040	0.103	0.040
L-Curve	5.143	1.791	9.300	4.036
LTN	5.297	2.162	8.078	3.071
Validation croisée	219.100	161.677	378.757	299.804

TABLE 5.3 – Tableau des temps de calculs des méthodes de sélection (Validation croisée, L-Curve, LTN) du coefficient de pondération parmi 100 hypothèses. Chaque optimisation de la pose est effectuée sur 10 itérations.

La méthode par validation croisée *leave-one-out* est quant-à-elle beaucoup plus lente mais elle reste concevable notamment lorsque la contrainte possède peu de termes d'erreur. Notons de plus qu'avec la forme approximée et non itérative du critère de validation croisée [Bartoli 2008, Farenzena 2008], la méthode devrait bénéficier d'un gain important en temps de calcul.

Notons que pour augmenter la rapidité de convergence des multiples optimisations, la pose initiale $\mathbf{P}_i^{(0)}(\lambda_i^l),\ \forall l \in [2, N_\lambda]$ sur le problème dont le poids est λ_i^l correspond à l'optimum estimé avec le problème précédent où le poids λ_i^{l-1} était légèrement inférieur au poids courant λ_i^l. Puisque les variations entre les poids sont faibles, l'optimisation converge rapidement (en moins d'itérations) vers une solution qui est proche de la solution précédente.

5.2.4 Ajustement de faisceaux local contraint

Ajustement de faisceaux local contraint. Lorsque tous les coefficients de pondération des différentes contraintes sont estimés, l'ajustement de faisceaux incrémental à l'étape i tente de minimiser une fonction mono-objectif (équation (5.13)) et les techniques de résolution de moindres carrés non-linéaires (voir la section 2.2.2) peuvent être employées. Le problème est le suivant :

$$\mathbf{x}_i^* = \arg\min_{\mathbf{x}_i} F_w^{LBA}(\mathbf{x}_i; \{\lambda_1, \ldots, \lambda_i\}). \tag{5.12}$$

où les pondérations $\lambda_{1..i-1}$ ont été sélectionnées lors des ajustements précédents et λ_i dans l'étape de sélection du poids précédente.

En pratique, nous avons opté pour l'algorithme de LEVENBERG-MARQUARDT qui a montré expérimentalement de bons résultats. La résolution creuse par bloc peut aussi être conservée dans la plupart des cas. Certaines contraintes, sur le déplacement notamment, réduisent légèrement la structure creuse de la matrice hessienne approximée, mais n'accroissent que légèrement le temps de résolution du problème.

Les poids sélectionnés ne sont pas remis en cause. Notons que dans ce contexte, nous ne remettons jamais en cause les poids des anciennes contraintes. Nous considérons qu'il n'est pas nécessaire de ré-estimer ceux-ci dans l'ajustement de faisceaux car cela entraînerait beaucoup de calcul. Nous pensons que les coefficients de pondération estimés sur les paramètres de pose peuvent être réutilisés directement dans l'ajustement de faisceaux local et qu'il n'est pas nécessaire de remettre en cause les anciens coefficients de pondération. En effet, la fonction de coût de l'ajustement de faisceaux peut être réécrite de la manière suivante :

$$F_w^{LBA}(\mathbf{x}_i; \{\lambda_1, \ldots, \lambda_i\}) = \underbrace{\varepsilon_{1..i-1}^{c+k}(\mathbf{x}_i)}_{\text{Erreurs et contraintes des images précédentes}} + \underbrace{\varepsilon_i^{c+k}([\mathbf{P}_i \mathbf{s}_i]) + \lambda_i^2 \varepsilon_i^k(\mathbf{P}_i)}_{\text{Erreurs et contrainte de la pose } i}. \tag{5.13}$$

Le second terme de la fonction de coût de l'ajustement de faisceaux local contraint correspond à la fonction de coût précédemment minimisée durant l'optimisation de la pose courante ((5.4)). À ce terme d'erreur nous ajoutons les erreurs des nouveaux points 3D générés lors de l'étape de triangulation effectuée entre l'estimation du poids courant et l'ajustement de faisceaux local. En effet, si l'on note l'ensemble des points 3D à l'étape i : $\mathbf{s}_i = [\mathbf{s}_i^{\text{old}} \mathbf{s}_i^{\text{new}}]$ où $\mathbf{s}_i^{\text{new}}$ est l'ensemble des nouveaux points 3D générés avant l'étape d'ajustement de faisceaux local contraint, alors l'erreur de reprojection moyenne ((5.2)) pour la pose courante i peut s'écrire :

$$\bar{\varepsilon}_i^c(\mathbf{x}_i) = \frac{1}{Card(j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{old}}) + Card(j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{new}})} \left(\sum_{j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{old}}} \nu_j^i d^2(\hat{\mathbf{q}}_j^i, \mathcal{P}_i \mathbf{Q}_j) + \sum_{j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{new}}} \nu_j^i d^2(\hat{\mathbf{q}}_j^i, \mathcal{P}_i \mathbf{Q}_j) \right). \tag{5.14}$$

En pratique, le nombre de points 3D nouvellement reconstruits est en moyenne le tiers du nombre de points 3D triangulés lors des poses clés précédentes : $\frac{Card(\mathbf{s}_i^{\text{new}})}{Card(\mathbf{s}_i^{\text{old}})} \approx \frac{1}{3}$. Ce ratio augmente lorsque la scène change rapidement, comme par exemple dans les virages. Nous faisons donc l'hypothèse que l'erreur moyenne de reprojection pour la pose i est peu modifiée avec l'intégra-

tion des nouveaux points 3D :

$$\bar{\varepsilon}_i^c(\mathbf{x}_i) \approx \frac{1}{Card(j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{old}})} \sum_{j|\mathbf{Q}_j \in \mathbf{s}_i^{\text{old}}} \nu_j^i d^2(\hat{\mathbf{q}}_j^i, \mathcal{P}_i \mathbf{Q}_j). \tag{5.15}$$

Ce terme d'erreur correspond à la première partie (vision) de la fonction de coût qui a été minimisée lors de la sélection du coefficient de pondération (5.4), mais où seuls les paramètres de la pose courantes ont été optimisés.

Notons que le facteur de pondération est estimé avant l'étape de triangulation, afin que les nouveaux points 3D reconstruits bénéficient de l'estimation contrainte de la pose.

5.3 Application au SLAM multi-capteur

Cette section présente comment la technique de fusion générique par ajustement de faisceaux peut être utilisée dans le cadre d'un SLAM visuel doté d'un second capteur mesurant les position ou déplacements du système.

5.3.1 Intégration dans un SLAM visuel existant

Lors de cette thèse, nous avons eu accès à l'implémentation en langage *C++* d'un SLAM monoculaire temps réel, dérivé des travaux de recherche d'ÉTIENNE MOURAGNON [Mouragnon 2007] et décrite succinctement dans la section 3.3.2. Cette implémentation nous a servie de base dans laquelle nous avons intégré nos travaux : l'intégration d'informations externes dans l'ajustement de faisceaux local d'un SLAM visuel.

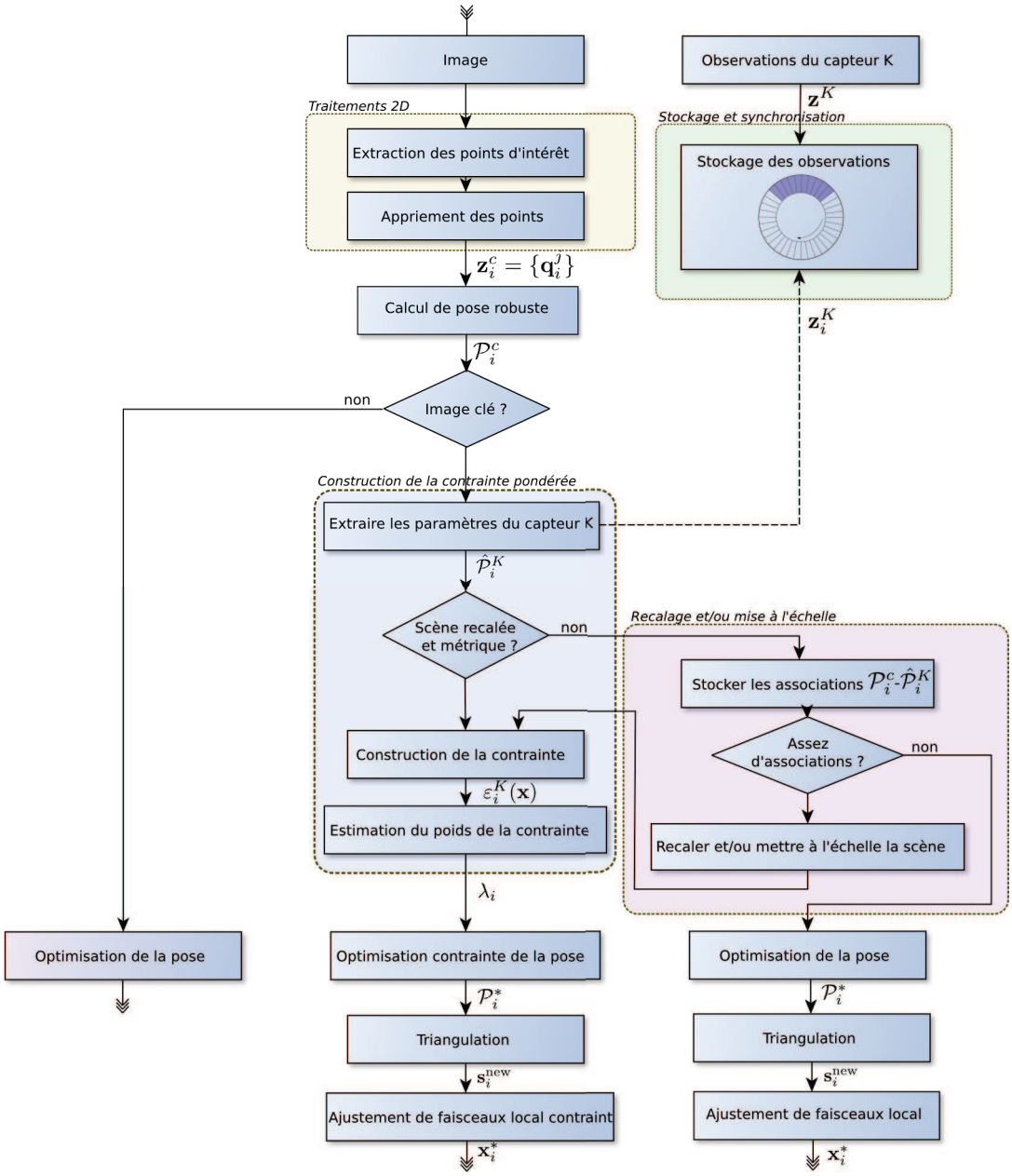


FIGURE 5.10 – Schéma de fonctionnement du SLAM avec un second capteur de mouvement.

La figure 5.10 illustre le processus incrémental du SLAM multi-capteur. Est intégré dans ce schéma la méthode de fusion de données sans incertitude connue par ajustement de faisceaux, correspondant à l'algorithme 10 présenté dans la section 5.2.

5.3.2 Aperçu du système

Nous résumons le SLAM multi-capteur par les étapes suivantes, où **(*)** présente une nouvelle étape ou une modification majeure du SLAM visuel original [Mouragnon 2007].

Localisation. (resection)

1. Détection et appariement des points d'intérêt (sections 3.1.1.1 et 3.1.1.3)
2. Estimation robuste de la pose par vision $\mathcal{P}_i^c$ (section 3.1.3)
3. **(*)** Extraction des observations du second capteur et prédiction de la pose du système $\hat{\mathcal{P}}_i^k$. Construction de la contrainte ε_i^k (section 5.3.4)
4. **(*)** Sélection du poids λ_i associé à la contrainte ε_i^k (section 5.2.3)
5. **(*)** Optimisation contrainte de la pose i (section 5.2.3.1)

Cartographie. (intersection et raffinement)

1. Trianguler les nouveaux points 3D (section 3.1.2)
2. **(*)** Ajustement de faisceaux local contraint (section 5.2.4).

Les section suivantes présentent chacune des nouvelles étapes.

Avant de pouvoir intégrer les observations du second capteur et appliquer la méthode générique de fusion de données par ajustement de faisceaux, plusieurs étapes sont nécessaires : intégration des données, changement de repère (par la matrice de calibrage capteur-caméra $\mathcal{C}_{c/k}$), prédiction à l'instant de l'image, choix de la contrainte et recalage dans un repère absolu (terrestre), ou encore mise à l'échelle métrique de la scène.

5.3.3 Recalage et mise à l'échelle métrique de la scène

Rappelons que le SLAM monoculaire délivre une reconstruction et une localisation à une similitude près. La scène est exprimée dans un repère local : la première caméra est à l'origine et son orientation est la matrice identité. De plus, l'échelle de la scène est de plus inconnue.

Certains capteurs comme le GPS ou les centrales à inertie, délivrent leurs observations exprimées dans un repère absolu, souvent le repère terrestre $\mathcal{W}^g$. Il peut être alors intéressant d'exploiter ces informations complémentaires afin de recaler la reconstruction dans ce repère absolu, beaucoup plus exploitable, notamment si l'on veut localiser un véhicule sur Terre et de coupler le système avec un SIG (Système d'Information Géographique) par exemple. La métrique de la reconstruction peut aussi être estimée si l'on utilise un odomètre ou tout autre capteur mesurant la position ou le déplacement métrique du système.

Le processus mis en œuvre pour recaler la reconstruction de la scène dans le repère fourni par le second capteur et la mise à l'échelle métrique de celle-ci est réalisé conjointement, en deux étapes : construction d'un historique de correspondances de poses (ou seulement une partie : position ou orientation) entre la pose estimée par le SLAM visuel (pose clé ${}^{\mathcal{W}_c}\mathbf{P}_i^c$) et celle observée par le second capteur dans le repère monde du capteur (${}^{\mathcal{W}_k}\hat{\mathbf{P}}_i^k$) : $h_{recal} = \{({}^{\mathcal{W}_c}\mathbf{P}_1^c, {}^{\mathcal{W}_c}\mathbf{P}_1^k); \dots ({}^{\mathcal{W}_k}\mathbf{P}_{N_{recal}}^c, {}^{\mathcal{W}_k}\mathbf{P}_{N_{recal}}^k); \}$. À partir de ces correspondances, nous estimons la similitude moyenne $\mathcal{H}$ à appliquer sur la scène. En pratique, au début de la séquence, le SLAM visuel pur est exécuté afin d'obtenir une première localisation, jusqu'au moment où nous considérons avoir un nombre suffisant de correspondances inter-pose ($N_{recal} = 6$) pour estimer précisément la métrique et le changement de repère local du SLAM vers le repère absolu.

Une fois que la similitude $\mathcal{H} = \begin{pmatrix} s\mathcal{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}$ est estimée par les correspondances de poses, nous recalons la scène et la trajectoire par la matrice $\mathcal{H}'$ de la manière suivante :

$$\forall i \in [1, N_m] \ : \ \mathbf{t}_i^c = \mathcal{H}'\mathbf{t}_i^c \ \text{ et } \ \mathcal{R}_i^c = \mathcal{R}\mathcal{R}_i^c \tag{5.16}$$

$$\forall j \in [1, N_n] \ : \ \mathbf{Q}_j = \mathcal{H}'\mathbf{Q}_j, \tag{5.17}$$

avec

$$\mathcal{H}' = \underbrace{\begin{pmatrix} \mathcal{I} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}}_{\text{translation}} \underbrace{\begin{pmatrix} \mathcal{I} & \mathcal{R}\mathbf{t}_{N_{recal}}^c \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} s\mathcal{I} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathcal{I} & -\mathcal{R}\mathbf{t}_{N_{recal}}^c \\ \mathbf{0}^\top & 1 \end{pmatrix}}_{\text{mise à l'échelle}} \underbrace{\begin{pmatrix} \mathcal{R} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{pmatrix}}_{\text{rotation}}. \tag{5.18}$$

5.3.3.1 Estimation de la similitude

La matrice de similitude est composée de 3 éléments : orientation, translation et facteur d'échelle. Chacun est estimé de la manière suivante.

Estimation du changement de repère relatif-terrestre. L'estimation du changement de repère local-terrestre, c'est-à-dire de la transformation euclidienne $(\mathcal{R} \ \mathbf{t})$ de la similitude $\mathcal{H}$, est réalisée directement en prenant la correspondance des poses la plus récente : $({}^{\mathcal{W}_c}\mathcal{T}_{N_{recal}}^c, {}^{\mathcal{W}_k}\mathcal{T}_{N_{recal}}^k)$, par la formule : $\begin{pmatrix} \mathcal{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} = ({}^{\mathcal{W}_c}\mathcal{T}_{N_{recal}}^c)^{-1}{}^{\mathcal{W}_k}\mathcal{T}_{N_{recal}}^k$.

Ce recalage présuppose que la dernière pose estimée par le capteur k est assez précise. Notons qu'il aurait été préférable de réaliser une moyenne sur l'ensemble des correspondances afin d'être beaucoup plus robuste à l'imprécision éventuelle de la dernière pose, par une méthode de *best-fit* sur les positions par exemple. Mais la précision de recalage dans le repère absolu n'est pas l'objectif de ces travaux puisque nous nous intéressons ici plus aux capteurs relatifs, mesurant le mouvement du système.

Estimation de l'échelle métrique Certaines observations permettent d'extraire l'information d'échelle, par exemple les capteurs mesurant les translations comme les odomètres.

Pour redimensionner la reconstruction et relier celle-ci à l'échelle métrique, nous calculons les normes de déplacement entre deux images clés à partir de la trajectoire donnée par le SLAM visuel et des prédictions effectuées par le second capteur de l'historique des correspondances des poses. Notons $s_i^k = \|\mathbf{dt}_i^k\|$ et $s_i^c = \|\mathbf{dt}_i^c\|$ ces normes de déplacement pour respectivement les déplacements estimés par le capteur k et la vision (c), pour les $N_{recal} - 1$ poses ($\forall i \in [2, N_{recal}]$). Les vecteurs $\mathbf{dt}_i$ sont les translations inter-caméra clé i : $\mathbf{dt}_i = \mathbf{t}_i - \mathbf{t}_{i-1}$.

Le facteur de redimensionnement s à appliquer à la reconstruction (et à la trajectoire) est alors le rapport des moyennes de chacun des capteurs :

$$s = \frac{\frac{1}{N_{recal}} \sum\limits_{i \in [2, N_{recal}]} s_i^k}{\frac{1}{N_{recal}} \sum\limits_{i \in [2, N_{recal}]} s_i^c}. \tag{5.19}$$

Remarques. Pour éviter que l'ajustement de faisceaux local ne modifie le recalage nous fixons la jauge de tous les ajustement de faisceaux en enlevant la dernière pose $\mathcal{P}_{N_{recal}}$ de l'optimisation, en gardant cependant les erreurs de reprojection de la fonction de coût. Notons que l'échelle n'est pas fixée dans les ajustements. Il aurait été concevable d'imposer par exemple la norme de déplacement entre les poses $N_{recal} - 1$ et N_{recal} afin de fixer l'échelle. En pratique, nous n'avons pas observé de variation importante de l'échelle (dans les ajustement locaux), si ce n'est le problème récurrent de dérive d'échelle due à la méthode de SLAM incrémental.

La qualité de cette méthode d'estimation du recalage et de mise à l'échelle de la scène suppose que d'une part, la localisation initiale par SLAM visuel pur est relativement précise : le mouvement initial du système ne doit pas être un mouvement critique pour les algorithmes de vision (rotation pure, etc.), et d'autre part que le second capteur est au début assez précis, sans dérive importante. Ces hypothèses de travail sont pour certains capteurs difficiles à tenir, notamment avec la centrale inertielle. En effet, nous avons tenté d'estimer le facteur d'échelle à l'aide des observations des accéléromètres de la centrale inertielle de faible coût XSENS-MTi (section 1.3.2.4) (par une double intégration temporelle des accélérations), mais l'importance du bruit des mesures et le temps nécessaire (> 10 secondes) à la construction de N_{recal} caméras clés, impliquent une dérive importante sur l'estimation de l'échelle. La technique n'est pas adaptée dans ces conditions. Nous n'employons pas pour cela les accélérations du capteur XSENS-MTi.

5.3.4 Construction des contraintes

5.3.4.1 Principe

Lorsqu'une nouvelle image clé est détectée, le processus de construction de la contrainte extrait d'une mémoire tampon circulaire, l'ensemble des observations produites par le second capteur ayant une date d'acquisition inférieure ou égale à la date t_i de l'image i. Chacune des observations est ensuite intégrée dans l'ordre chronologique afin de construire une estimation des paramètres de pose ou seulement une composante. Un modèle de mouvement est ensuite utilisé afin de prédire ces paramètres à la date précise de l'image clé. En effet, si l'on veut

contraindre précisément dans l'ajustement de faisceaux la pose i, il est nécessaire de synchroniser les observations du capteur k avec les images clés. Enfin, la contrainte est construite à partir de cette prédiction et de la pose estimée par la vision. La figure 5.11 illustre l'acquisition temporelle des mesures des capteurs.

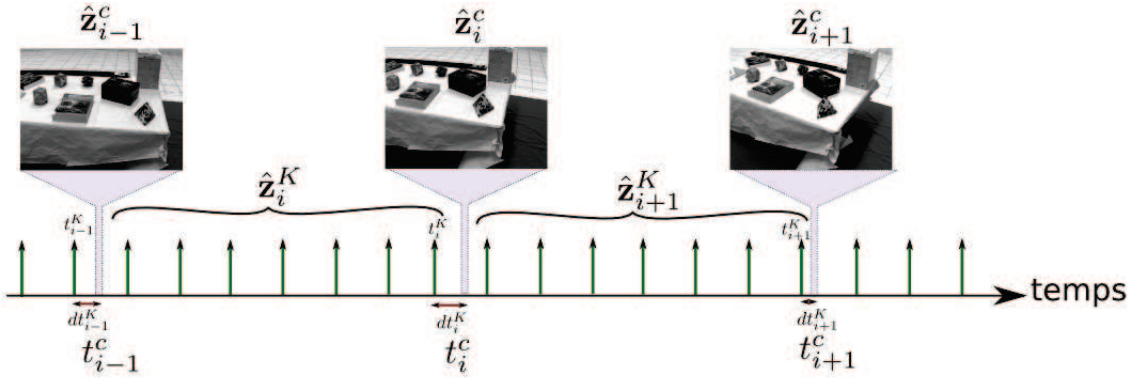


FIGURE 5.11 – Illustration de l'acquisition asynchrone des observations provenant du capteur caméra c et du capteur k.

5.3.4.2 Intégration des observations

L'objectif ici est de construire une pose prédite par le second capteur k à partir des observations notées $\mathbf{z}^k_i$, acquises entre les images clés précédente $i-1$ et courante i. Pour cela la méthode se compose de trois étapes :

- Intégration temporelle ou sélection de la dernière donnée
- Prédiction de la pose à l'instant précis de l'image clé : $X_{dt_i}(\mathbf{x}_i)$ (voir section 1.2.1)
- Changement du repère local capteur vers le repère caméra $\mathcal{C}_{k/c}$

L'intégration des observations varie suivant l'origine du second capteur k. Les capteurs utilisés acquièrent les observations de manière synchrone, à fréquence f^k fixe. L'intervalle de temps entre deux mesures est noté $l^k = \frac{1}{f^k}$. Notons de plus $dt^k_i = t^c_i - t^k_i$, le temps entre l'instant (*timestamp*) de l'image clé i et l'instant de la dernière observation du capteur k (t^k_i), mesurée juste avant l'acquisition de l'image clé i.

Nous présentons dans la suite comment sont intégrées les observations des 3 capteurs utilisés dans cette thèse : l'odomètre, la centrale inertielle XSENS-MTi et le système de navigation à guidage inertiel IXSEA-PHINS.

Odomètre. L'odomètre employé délivre à une fréquence de $f^o = 25$Hz une information de vitesse linéaire instantanée $\hat{v}^o_t$. À l'instant de l'image clé i, nous construisons le vecteur d'obser-

vations ${}^{\mathcal{W}_c}\hat{\mathbf{z}}_i^o = {}^{\mathcal{W}_c}\mathbf{dt}_i^o$ formé de l'estimation de la translation par la formule :

$$ {}^{\mathcal{W}_c}\mathbf{dt}_i^o = \underbrace{{}^{\mathcal{W}_c}\mathcal{T}_{i-1}^c}_{\text{Pose précédente}} \underbrace{\left(\prod_{t \in]t_{i-1}^c, t_i^o[} \begin{pmatrix} \mathcal{I} & \hat{v}_t^o l^o \mathbf{dir} \\ \mathbf{0}^\top & 1 \end{pmatrix} \right)}_{\text{Obs. inter-caméras clés}} \underbrace{\begin{pmatrix} \mathcal{I} & \hat{v}_{t_i^o}^o dt_i^o \mathbf{dir} \\ \mathbf{0}^\top & 1 \end{pmatrix}}_{\text{Prédiction}}, \tag{5.20} $$

où $\mathbf{dir} = [0\ 0\ 1]^\top$ est la direction de déplacement dans le repère de la caméra (la matrice de calibrage entre l'odomètre et la voiture est l'identité $\mathcal{W}_{o/c} = \mathcal{I}$). Cette mesure sera ensuite utilisée d'un part pour estimer le facteur d'échelle métrique de la reconstruction, puis comme contrainte directement dans l'ajustement de faisceaux, sur l'échelle ou sur la translation.

Centrale inertielle (XSENS-MTi). Avec la centrale inertielle XSENS-MTi, nous utilisons seulement l'observation d'orientation la plus récente mesurée (orientation $\hat{\mathcal{R}}_{t_i^x}^x$ et vitesse angulaire $\hat{\boldsymbol{\omega}}_{t_i^x}^x$). Le vecteur de mesure est constitué de l'orientation $\hat{\mathbf{z}}_i^x = \text{vect}(\mathcal{R}_i^x)$, tel que

$$ {}^{\mathcal{W}_x}\mathcal{R}_i^x = \underbrace{{}^{\mathcal{W}_x}\hat{\mathcal{R}}_{t_i^x}^x}_{\text{Dernière Obs.}} \underbrace{\text{rot}_\text{e}\left(dt_i^x \hat{\boldsymbol{\omega}}_{t_i^x}^x\right)}_{\text{Prédiction}} \underbrace{\mathcal{R}_{x/c}}_{\text{Mat. de calibrage IMU-caméra}}. \tag{5.21} $$

Le changement de repère local entre le capteur caméra et le capteur k est effectué par la matrice $\mathcal{C}_{x/c}$. Cette matrice a été estimée par une procédure de calibrage s'apparentant aux travaux de LOBO *et al.* dans [Lobo 2007] (seule la matrice de rotation a été ici calculée).

Les informations d'accélération ne sont pas utilisées pour estimer l'échelle de la scène, ou pour contraindre l'ajustement de faisceaux en position/translation/échelle. En effet, la méthode que nous proposons consiste à contraindre chaque image clé de la reconstruction et non toutes les images. Or, le temps entre deux images clés peut être assez important (plusieurs secondes) et, lorsque l'on intègre les accélérations bruitées, l'estimation du déplacement est trop peu précise pour pouvoir être utilisée. Les bruits des mesures d'accélération (voir la section 1.3.2.2 page 27) proviennent de plusieurs sources : approximation du calibrage des accéléromètres (biais $\mathbf{b}$ des alignements approximatifs des axes), ainsi que de la qualité de la suppression de la gravité terrestre des mesures d'accélération. En effet, la qualité de la correction dépend principalement de la précision de l'orientation du capteur dans le repère terrestre. Avec la technologie MEMS, cette orientation n'est généralement pas assez précise pour pouvoir estimer précisément la position du capteur : la gravité n'est pas intégralement supprimée et la double intégration temporelle des accélérations entraîne rapidement une dérive importante de vitesse et de position. À l'image des travaux proposés par YOU *et al.* [You 2001], nous fusionnons les orientations (issues de l'intégration des mesures des gyroscopes par la centrale inertielle) dans le cadre d'une localisation par SLAM avec ajustement de faisceaux (et non par filtre de Kalman étendu dans une localisation par cibles).

Système de navigation à guidage inertiel (INS-IXSEA-PHINS). Le système de navigation à guidage inertiel IXSEA-PHINS (introduit dans la section 1.3.3.2 page 30) peut quant-à-lui

être utilisé pour recaler la scène dans le repère terrestre et estimer la métrique de la reconstruction puisque les observations de position $\hat{\mathbf{t}}_t^n$ et d'orientation $\hat{\mathcal{R}}_t^n$ sont exprimées dans ce repère (excepté en intérieur où le GPS ne peut être utilisé). Le vecteur de mesure est à l'image de la centrale inertielle XSENS-MTi, construit à partir de l'observation la plus récente ${}^{\mathcal{W}_n}\hat{\mathbf{z}}_i^n = [\text{vect}({}^{\mathcal{W}_n}\mathcal{R}_i^n)\ {}^{\mathcal{W}_n}\mathbf{t}_i^n]$, avec

$$ {}^{\mathcal{W}_n}\hat{\mathcal{T}}_i^n = \begin{pmatrix} {}^{\mathcal{W}_n}\hat{\mathcal{R}}_i^n & {}^{\mathcal{W}_n}\hat{\mathbf{t}}_i^n \\ \mathbf{0} & 1 \end{pmatrix} = \underbrace{{}^{\mathcal{W}_n}\hat{\mathcal{T}}_{t_i^n}^n}_{\text{Dernière Obs.}} \underbrace{\begin{pmatrix} \text{rot}_\text{e}\left(dt_i^n \boldsymbol{\omega}_{t_i^n}^n\right) & dt_i^n \mathbf{dt}_{t_i^n}^n \\ \mathbf{0}^\top & 1 \end{pmatrix}}_{\text{Prédiction}} \underbrace{\mathcal{C}_{n/c}}_{\text{Mat. de calibrage INS-caméra}}, \tag{5.22} $$

où $\mathbf{dt}_{t_i^n}^n$ et $\boldsymbol{\omega}_{t_i^n}^n$ sont des approximations des vitesses linéaire et angulaire, estimées à l'aide de la mesure précédente $\hat{\mathcal{T}}_{t_i^n - l^n}^n$:

$$ \begin{cases} \boldsymbol{\omega}_{t_i^n}^n = \frac{1}{l^n} \text{rot}_\text{e}^{-1}\left((\hat{\mathcal{R}}_{t_i^n - l^n}^n)^\top \hat{\mathcal{R}}_{t_i^n}^n \right) \\ \mathbf{dt}_{t_i^n}^n = \frac{1}{l^n} (\hat{\mathcal{R}}_{t_i^n - l^n}^n)^\top (\hat{\mathbf{t}}_{t_i^n}^n - \hat{\mathbf{t}}_{t_i^n - l^n}^n) \end{cases} . \tag{5.23} $$

La qualité des mesures produites par la technologie par fibre optique employée dans la centrale IXSEA-PHINS permet d'utiliser la centrale pour estimer l'échelle de la trajectoire même en intérieur, mais il faut le faire au début de la séquence, c'est-à-dire lorsque la centrale dérive peu. En effet celle-ci, sans recalage extérieur (GPS ou autres), peut dériver en position de 3 mètres en 100 secondes ($0.3mm/s^2$). Bien que cette dérive soit bien inférieure à la dérive en position de la centrale XSENS-MTi ($20mm/s^2$), sans recalage extérieur, la centrale IXSEA-PHINS ne peut pas vraiment être intégrée dans notre approche afin de contraindre l'ajustement de faisceaux en position/translation/échelle. Elle peut parfaitement en revanche être utilisée pour contraindre les orientations.

5.3.4.3 Choix de la contrainte

Une fois que le vecteur d'observations synchronisées du second capteur $\hat{\mathbf{z}}_i^k$ a été construit à partir des observations inter-caméras clés, la contrainte peut être générée. Il s'agit généralement d'une erreur mesurant la distance entre les paramètres estimées par le capteur k et ceux estimés par la vision. Bien entendu les paramètres contraints dépendent du capteur utilisé, des éléments de $\hat{\mathbf{z}}_i^k$. Par exemple la centrale inertielle peut être utilisée pour contraindre l'orientation de la caméra ou la rotation effectuée depuis la dernière caméra clé.

Des exemples de contraintes sont présentées dans la section 5.2.1.2.

5.4 Résultats expérimentaux

Nous avons expérimenté la fusion multi-capteur sans certitude par ajustement de faisceaux dans deux types d'applications : la localisation de véhicule, dans laquelle un odomètre placé dans les roues mesure la vitesse linéaire (translation) de déplacement du véhicule, et dans le cadre d'un

mouvement libre dans lequel un opérateur déplace le système muni d'une caméra et soit d'une centrale inertielle, soit d'un système de navigation à guidage inertiel. Dans ces conditions, les localisations estimées par les algorithmes de SLAM monoculaire sont souvent affectées par des dérives dues à l'incrémentalité de l'algorithme. Nous pensons que le SLAM par ajustement de faisceaux contraint peut réduire ces dérives grâce à l'utilisation de capteurs de mouvement.

5.4.1 Préliminaires

Dans les séquences présentées dans cette section, nous comparons les qualités de la localisation du système obtenues par les trois différentes méthodes de sélection de poids. Pour cela, nous mesurons les erreurs de localisation de chaque méthodes par rapport à une trajectoire précise *vérité terrain* provenant soit d'un laser de poursuite, soit d'un système de navigation à guidage inertiel couplé à un GPS RTK. Ces systèmes sont présentés dans les sections 1.3.3.3, 1.3.3.2 et 1.3.3.1.

Mesure de la qualité de la localisation. La qualité de la localisation par rapport à la vérité terrain est notamment mesurée par la moyenne des résidus 3D $\mu_{\text{3D}}(\mathbf{x})$:

$$\mu_{\text{3D}}(\mathbf{x}) = \frac{1}{N_m} \sum_{i=1}^{N_m} d({}^{\mathcal{W}_{gt}}\mathbf{t}_i, {}^{\mathcal{W}_{gt}}\check{\mathbf{t}}_i), \tag{5.24}$$

où ${}^{\mathcal{W}_{gt}}\check{\mathbf{t}}_i$ sont les positions de la vérité terrain à l'instant t_i de l'image i exprimées dans le repère de la vérité terrain ($\mathcal{W}_{gt}$, qui peut être le repère monde $\mathcal{W}_g$). Puisque les temps des données ne correspondent pas exactement, nous utilisons une interpolation temporelle linéaire entre la donnée précédant le temps t_i de l'image clé et la données suivante.

De plus, il est fréquent d'avoir des données de vérité terrain exprimées dans un repère inconnu différent des données de localisation du SLAM multi-capteur. L'échelle de la localisation du SLAM peut aussi ne pas être métrique. Pour pouvoir comparer ces informations nous procédons à un recalage (et à une mise à l'échelle lorsque cela est nécessaire, c'est-à-dire pour les localisations par SLAM monoculaire seul ou avec une centrale inertielle) en réalisant un *best-fit* sur les 5 premières positions non alignées de la caméra. En effet, les données de la vérité terrain ne concernent que la position du système, sans l'orientation. Ce recalage permet d'exprimer les positions des poses clés dans le repère de la vérité terrain ($\mathcal{W}_{gt}$).

Mesure de la qualité de l'échelle. Dans les expérimentations sur la localisation de véhicule où un odomètre est présent, nous mesurons la qualité de l'échelle de la localisation. Pour cela, nous calculons les statistiques (moyenne, médiane, écart-type, maximum) sur les ratios des normes des translations. La moyenne $\mu_{\text{scale}}(\mathbf{x})$ est définie par :

$$\mu_{\text{scale}}(\mathbf{x}) = \frac{1}{N_m - 1} \sum_{i=2}^{N_m} \frac{d({}^{\mathcal{W}_{gt}}\mathbf{t}_i, {}^{\mathcal{W}_{gt}}\mathbf{t}_{i-1})}{d({}^{\mathcal{W}_{gt}}\check{\mathbf{t}}_i, {}^{\mathcal{W}_{gt}}\check{\mathbf{t}}_{i-1})}. \tag{5.25}$$

Lorsque l'échelle métrique de la localisation est bien estimée et conservée tout au long de la séquence, la moyenne $\mu_{\text{scale}}(\mathbf{x})$ doit être égale à 1.

Configuration de la caméra. Les séquences réelles ont été enregistrées avec une caméra perspective monochrome de marque AVT GUPPY F-046B (capteur 1/2"). L'optique utilisée est une optique à large champs de vue (focale de 4mm). Cette caméra enregistre à une fréquence de 30 images par secondes des images 640 x 480 pixels monochromes. Le modèle mathématique de la caméra est décrit dans la section 1.3.1.

5.4.2 Localisation d'un véhicule

La première expérimentation concerne la localisation d'un véhicule, où un odomètre placé dans les roues du véhicule est utilisé afin de réduire la dérive d'échelle du SLAM monoculaire.

5.4.2.1 Présentation

Le projet de recherche ANR Odiaac, dont le consortium était composé de Renault, Dotmobil, du LASMEA et du CEA LIST (laboratoire LVIC : vision et ingénierie des contenus), avait pour objectif de localiser un véhicule en temps réel en milieu urbain en utilisant un ensemble de capteurs bon marché : GPS, odomètres et caméra monoculaire calibrée (AVT GUPPY F-046B). Un système de navigation à guidage inertiel (IXSEA-LANDINS) couplé à un GPS RTK, aussi appelé trajectomètre, est intégré au véhicule et a servi de vérité terrain. La précision annoncée de ce système est de 50 cm, mais en pratique nous avons observé à l'aide d'un plan de ville, une erreur moyenne proche du mètre. La séquence `Odiaac` est issue d'une campagne d'acquisition réalisée à Saint Quentin en Yvelines (78) dans un trafic urbain. Trois images de la séquence `Odiaac` sont présentées dans la figure 5.12 et le tableau 5.4 mentionne les statistiques de la reconstruction par un algorithme de SLAM monoculaire [Mouragnon 2006].

(a) (b) (c)

FIGURE 5.12 – Images issues de la séquence `Odiaac`.

	Longueur *(m)*	**Nombre d'images (images clés)**	**Nombre de points 3D**
`Odiaac`	130	2500 (193)	6290

TABLE 5.4 – Statistiques de localisation et reconstruction d'un SLAM monoculaire sur la séquence `Odiaac`.

5.4.2.2 Résultats

Nous avons expérimenté la technique de fusion de données par ajustement de faisceaux contraint, en ajoutant dans la fonction de coût une contrainte sur la norme du déplacement inter-caméra clé. Ce terme de coût supplémentaire va avoir pour effet de contraindre l'échelle locale de la trajectoire grâce à l'information odométrique. La contrainte, notée $\varepsilon_i^{o,s}(\mathbf{x})$, est décrite dans le tableau 5.5, où le capteur k est l'odomètre o : $k = o$. Les méthodes de sélection des poids ont été effectuées avec 100 hypothèses de poids, dont les valeurs sont générées logarithmiquement (voir la section 5.2.3) de $\lambda^{min} = 10^{-3}$ à $\lambda^{max} = 10^{3}$.

Dans la séquence `Odiaac`, le véhicule a réalisé le tour d'un rond-point, de forme "étirée" (voir la figure 5.13). Les localisations estimées (et recalées dans le repère du trajectomètre) avec les différentes méthodes de sélection dynamiques des pondérations sont présentées dans la figure 5.13. Le tableau 5.13 mentionne les statistiques sur les localisations.

	médiane *(m)*	**moyenne** *(m)*	**écart-type** *(m)*	**max** *(m)*	**min** *(m)*
SLAM monoculaire	11.649	12.82	8.828	33.002	0.116
SLAM + CV_{loo}	2.043	**2.835**	2.055	7.437	0.283
SLAM + $L-Curve$	3.011	4.443	3.354	11.708	0.859
SLAM + LTN	5.174	6.663	4.703	12.755	0.775

TABLE 5.5 – Statistiques de la localisation sur la séquence `Odiaac` par rapport à la vérité terrain.

Localisation 3D. Si l'on compare les résultats du SLAM monoculaire seul (points rouges) par rapport au trajectomètre (points violets), on peut observer une dérive de position et d'échelle de la localisation importante : le rond-point reconstruit est plus petit que le rond-point réel. Dans l'ensemble, les trois méthodes améliorent significativement la qualité de la localisation du véhicule et diminuent en moyenne d'un facteur 2 à 5 l'erreur de positionnement 3D par rapport au trajectomètre. Notons que la méthode de sélection par validation croisée est plus efficace dans cette expérience que les deux autres méthodes de sélection de poids. La dérive en position et en échelle est assez bien corrigée par la sélection par validation croisée mais reste tout de même apparente avec les méthodes de sélection par compromis. On peut aussi remarquer une légère dérive en orientation dans la trajectoire estimée par le SLAM contraint avec la sélection de poids pas LTN, qui n'existait pas dans le SLAM monoculaire. La figure 5.14 représente l'évolution de l'erreur de localisation 3D en fonction du temps.

Sur cette figure, nous observons la forte croissance de l'erreur 3D du SLAM monoculaire non contraint, à cause d'une dérive d'échelle qui semble-t-il intervient approximativement 35

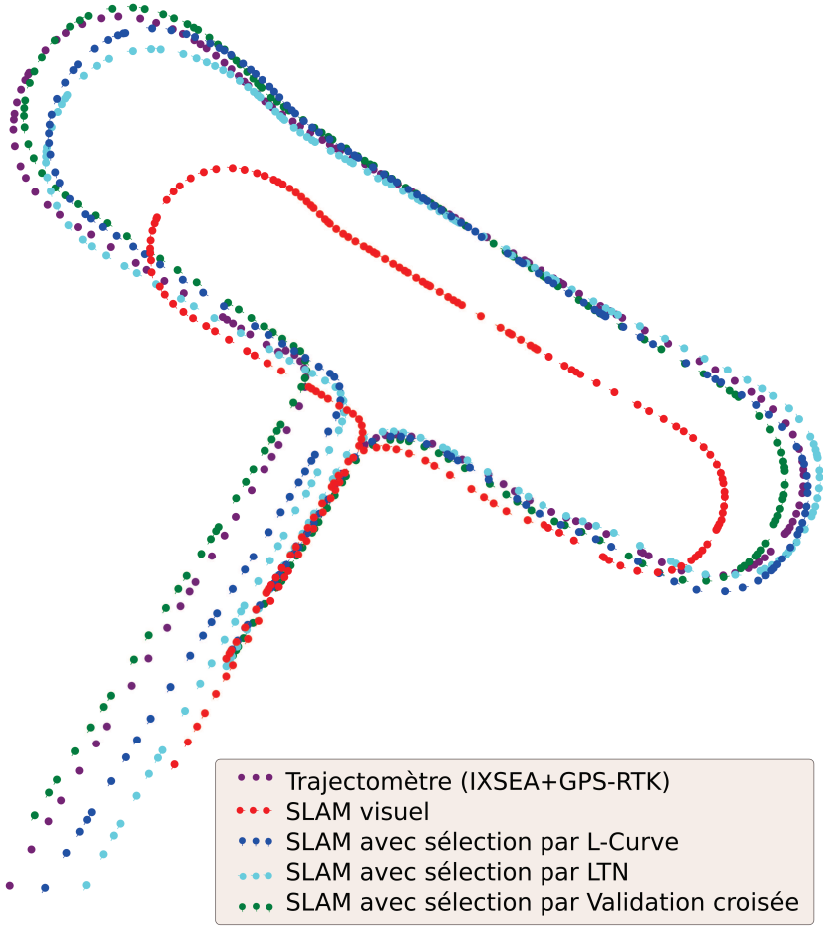


FIGURE 5.13 – Localisations par SLAM obtenues sur la séquence `Odiaac`.

secondes après le début de la séquence.

Évaluation de l'échelle. Si l'on regarde l'estimation de la norme de déplacement inter-caméra clé, qui inclue l'échelle locale de la trajectoire, on peut observer plus précisément la contraction de l'échelle de la trajectoire estimée par le SLAM monoculaire non contraint (courbe rouge) sur la figure 5.15. La dérive apparaît dès les 10 premières secondes. Cette dérive est en partie supprimée grâce à l'ajout de la contrainte sur l'échelle avec la sélection de poids par validation croisée (moyenne de 1.0177). Pour les méthodes de sélection par critère de compromis la dérive est réduite mais toujours présente surtout à la fin de la séquence, ce que confirme le tableau reprenant les statistiques de l'échelle de la trajectoire (tableau 5.6).

Sur cette séquence, la méthode de sélection des poids la plus efficace semble être la méthode par validation croisée *leave-one-out* puisque la moyenne du ratio des normes des déplacements inter-caméras clés est la plus près de l'unité, et que le ratio minimum est bien supérieur au minimum des 2 autres méthodes. Les deux autres méthodes corrigent néanmoins assez efficacement la dérive d'échelle (0.97 pour les deux méthodes).

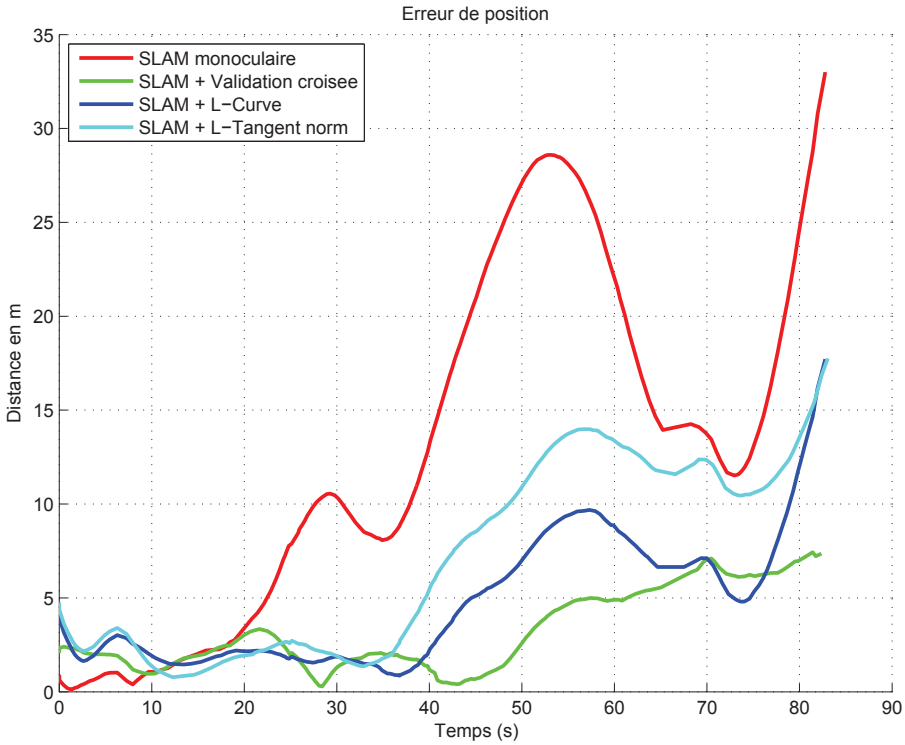


FIGURE 5.14 – Localisations par SLAM obtenues sur la séquence `Odiaac`.

	médiane	**moyenne**	**écart-type**	**max**	**min**
SLAM monoculaire	0.7689	0.808	0.144	2.112	0.590
SLAM + CV_{loo}	0.995	**1.017**	0.094	2.020	0.766
SLAM + $L-Curve$	0.954	0.973	0.145	2.376	0.699
SLAM + LTN	0.940	0.969	0.145	2.389	0.742

TABLE 5.6 – Statistiques sur le ratio des normes des translations (échelles) des trajectoires sur la séquence `Odiaac` par rapport à la vérité terrain.

Sélection des coefficients de pondération. La figure 5.16 montre les poids de chaque contrainte, sélectionnés par les 3 méthodes de sélection. Les statistiques de ces sélections sont énumérées dans le tableau 5.7.

	médiane *(m)*	**moyenne** *(m)*	**écart-type** *(m)*	**max** *(m)*	**min** *(m)*
CV_{loo}	3.8535	20.8030	28.0596	91.1163	0
$L-Curve$	0.0174	24.3322	40.3680	91.1163	0
LTN	0.0159	0.0225	0.0209	0.1963	0

TABLE 5.7 – Statistiques des poids sélectionnés sur la séquence `Odiaac`.

Nous pouvons remarquer que le choix des pondérations fluctue beaucoup entre les différents contraintes (présence de fortes variations). Cela signifie que la contrainte n'améliore pas toujours

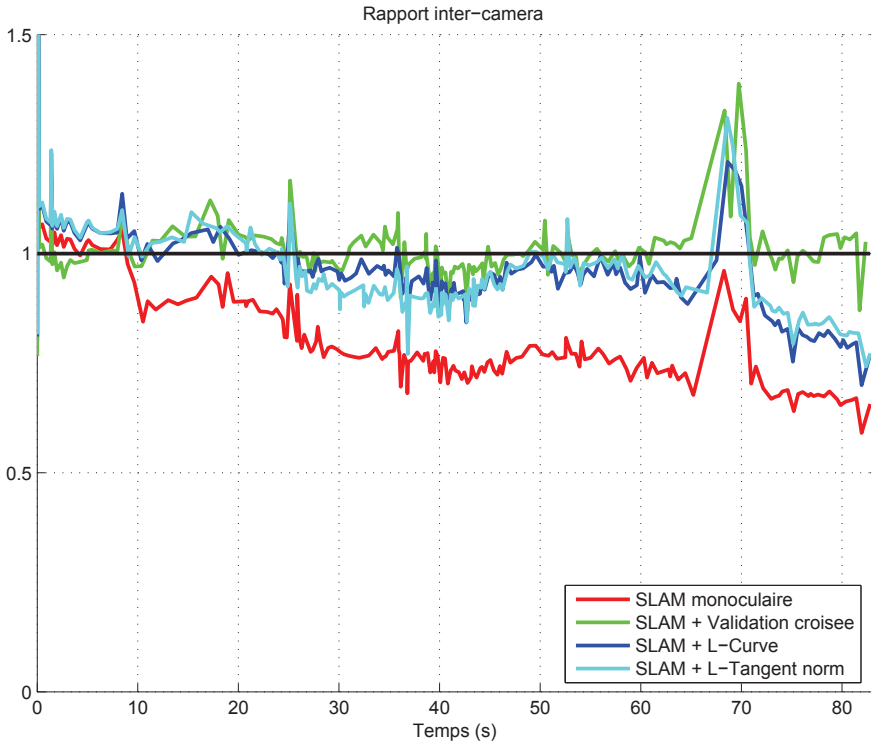


FIGURE 5.15 – Moyenne des carrés des résidus 3D sur la séquence `Odiaac` avec une contrainte d'échelle.

la localisation. D'autre part, les critères sur le compromis choisissent en moyenne des coefficients assez faibles, souvent aux alentours de 0.016 (médiane), alors que la méthode de validation croisée sélectionne une valeur de pondération proche de 4 (et 21 en moyenne). Les deux types de méthodes ne sont donc pas équivalentes.

Conclusion. Dans l'ensemble, l'ajout des contraintes d'orientation dans les ajustement de faisceaux locaux du SLAM améliore significativement la qualité de la localisation finale. Grâce à cela, la dérive d'échelle observée avec un SLAM non contraint est en partie supprimée, notamment grâce à la sélection des poids par validation croisée. Les méthodes de sélection par critères de compromis diminuent aussi la dérive d'échelle mais en conservent un partie (à la fin de la séquence), ce qui fait que l'erreur 3D de localisation finale est supérieure à l'erreur obtenue par la méthode de validation croisée. Ces résultats tendent à monter que la sélection par validation croisée *leave-one-out* est la méthode de sélection la plus adaptée à la fusion avec un odomètre.

5.4.3 Localisation en mouvement libre

La deuxième expérimentation a pour finalité de localiser une personne se déplaçant à l'intérieur d'un bâtiment. L'objectif est d'améliorer la qualité de la trajectoire estimée par un SLAM visuel en utilisant une centrale inertielle ou un système de navigation à guidage inertiel.

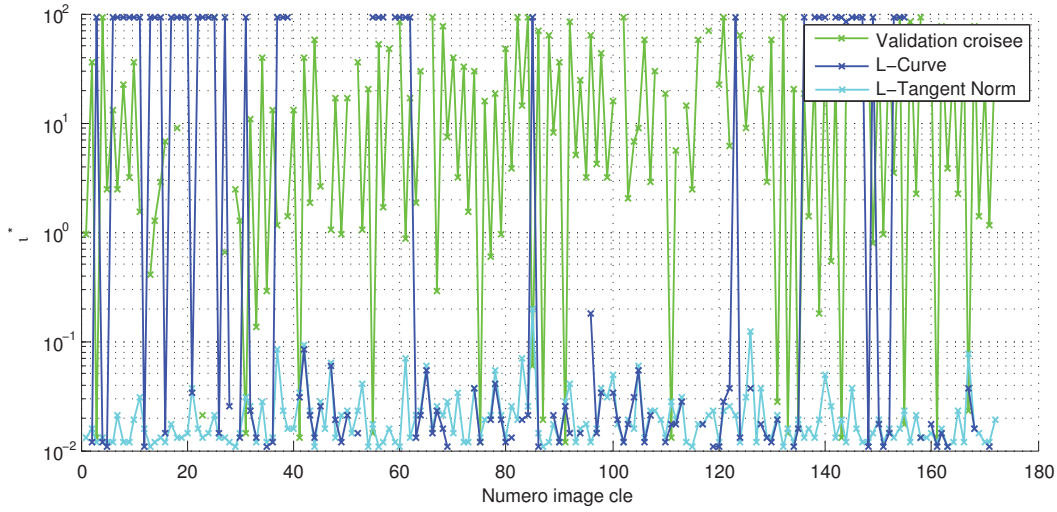


FIGURE 5.16 – Évolution des poids sélectionnés par les méthodes de validation croisée, L-Curve et LTN, avec une contrainte sur la rotation inter-caméras clés.

5.4.3.1 Localisation avec un système de navigation

Présentation du projet Gyroviz. La séquence `Gyroviz` est issue du projet de recherche éponyme. Ce projet de recherche ANR (Agence Nationale de la Recherche) regroupe différents acteurs industriels et de recherche :

- SOFRESUD, une entreprise spécialisée dans le développement de systèmes de défense,
- CEA-LIST, le laboratoire dans lequel j'ai effectué cette thèse,
- INRIA (GEOMETRICA), le groupe de recherche de l'INRIA spécialisé dans la géométrie algorithmique,
- IXSEA, une entreprise concevant des systèmes de navigation à guidage inertiel par fibre optique (notamment la centrale PHINS décrite en section 1.3.3.2),
- SUPMECA, l'institut supérieur de mécanique de Paris.

L'objectif du projet Gyroviz est la reconstruction 3D automatique de scènes à partir d'images localisées. Une illustration du système et de ses applications est donnée dans la figure 5.17. L'approche considérée dans ce projet est double : localiser en temps réel les images acquises par un appareil photographique ou un caméra professionnelle, et reconstruire de façon éparse (points 3D) mais en temps réel la scène observée avec une caméra basse résolution. Et d'autre part, de reconstruire en haute résolution texturée (*mesh*) la scène par une étape hors ligne.

La contribution du CEA LIST concerne la première partie de l'approche, c'est-à-dire la production d'images localisées en temps réel à l'aide d'une technique de SLAM monoculaire. Afin d'être robuste aux difficultés rencontrées avec la localisation monoculaire par *Structure-from-Motion* incrémental (image occultée, flou de vitesse, mouvements critiques, dérives, etc.), un système de navigation (IXSEA-PHINS) basé sur des capteurs inertiels à fibre optique est intégré au système. Ce système de navigation délivre des informations déjà intégrées (orientation et position) et présente donc des dérives temporelles lorsqu'il n'est pas recalé régulièrement par un élément extérieur (GPS, ...). Dans cette thèse, nous avons réalisé un système de SLAM visuel

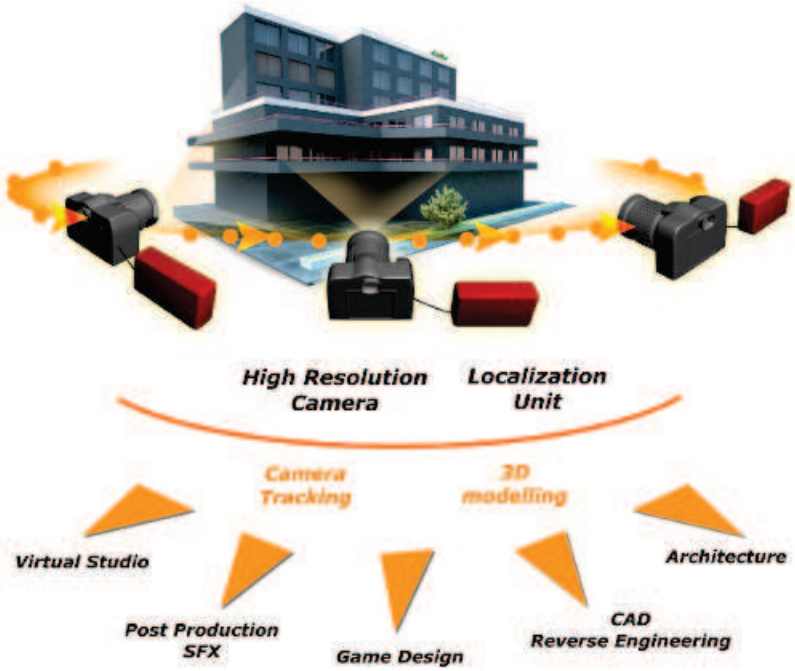


FIGURE 5.17 – Illustration du projet Gyroviz.

temps réel qui intègre les informations provenant de ce système dans le processus d'ajustement de faisceaux par la technique décrite au début de ce chapitre. Nous présentons ici les résultats obtenus par un traitement (SLAM par ajustement de faisceaux contraint) réalisé en temps différé mais en temps réel, après l'acquisition des données.

La séquence `Gyroviz` est issue d'une campagne d'acquisition réalisée dans les locaux de l'IMQ (Institut Méditerranéen de la Qualité), à Toulon. Quelques images de la séquence sont présentées dans la figure 5.18. La maquette d'essai comportent les capteurs fixés rigidement à la maquette suivants :

– une caméra AVT GUPPY F-046B monochrome calibré qui enregistre des images de résolution 640x480 pixels à une fréquence de 30 images par seconde (voir la section 1.3.1).
– une centrale de navigation IXSEA-PHINS (voir la section 1.3.3.2).

Le tableau 5.8 montre les statistiques de la localisation et reconstruction 3D par SLAM monoculaire sur la séquence `Gyroviz`.

	Longueur *(m)*	**Nombre d'images (images clés)**	**Nombre de points 3D**
`Gyroviz`	8.28	1360 (75)	1573

TABLE 5.8 – Statistiques de la localisation par SLAM sur la séquence `Gyroviz`.

Un système de laser de poursuite (voir la section 1.3.3.3) est employé afin de mesurer la trajectoire du système avec une précision importante (annoncée de $25\mu m$). Un opérateur muni de la maquette dirige la caméra vers la scène constituée d'éléments divers posés sur une table, et se déplace autour de la table avec un mouvement fluide sur approximativement 8 mètres, pendant 1

(a) (b) (c)

FIGURE 5.18 – Images issues de la séquence `Gyroviz`.

minute environ. Les observations asynchrones de chaque capteur (caméra, centrale de navigation et laser de poursuite) sont enregistrées durant le déplacement de l'opérateur. Les mesures sont datées sur une horloge commune afin de pouvoir fusionner et comparer toutes ces informations. Nous avons ensuite appliqué le SLAM visuel par ajustement de faisceaux contraint dont les contraintes sont réalisées à partir des informations d'orientation de la centrale de navigation.

Étant donnée que la centrale de navigation dérive en position de manière importante durant les expérimentations en intérieur (pas de recalage par GPS), nous avons choisi d'introduire dans l'ajustement de faisceaux des contraintes sur l'orientation, et plus précisément sur la rotation inter-caméras clés (la contrainte $\varepsilon_i^{k,dR}$ du tableau 5.1).

Comparaisons des localisations. Nous comparons les résultats des localisations par SLAM contraint avec une méthode combinant un SLAM visuel et un filtrage *a posteriori* par un filtre de Kalman étendu. Nous avons implémenté un filtre de Kalman étendu qui utilise la pose estimée par le SLAM en prédiction et fusionne les informations d'orientation (quaternions) issues du système de navigation. L'implémentation a été conçue à partir de l'article de KONOLIGE *et al.* [Konolige 2007] mais avec d'importantes modifications. D'une part, nous n'utilisons qu'un seule caméra (monoculaire, et non stéréo), de plus, le système de navigation utilisé ne délivrant pas d'information sur la gravité (direction/norme), nous n'avons pas intégré ces informations dans le filtre de Kalman. Seules les orientations du système de navigation sont ainsi intégrées dans le filtrage.

Les trajectoires estimées par les SLAM monoculaire non contraint, contraint avec la sélection par validation croisée et SLAM monoculaire puis fusionné *a posteriori* par un filtre de Kalman sont présentées dans la figure 5.19. Les deux autres méthodes de sélection ne sont pas représentés car les trajectoires sont similaires à la méthode par validation croisée.

La figure 5.20 et le tableau 5.9 présentent les erreurs de localisation 3D par rapport à la vérité terrain (laser de poursuite) pour les différents SLAM (monoculaire seul, contraint avec les 3 méthodes de sélection dynamique des poids et le SLAM couplé à un filtre de Kalman étendu *a posteriori*).

Les résultats sur les erreurs de localisation montrent que la méthode de SLAM par ajustement de faisceaux contraint améliore la qualité de la localisation lorsque l'on ajoute des informations

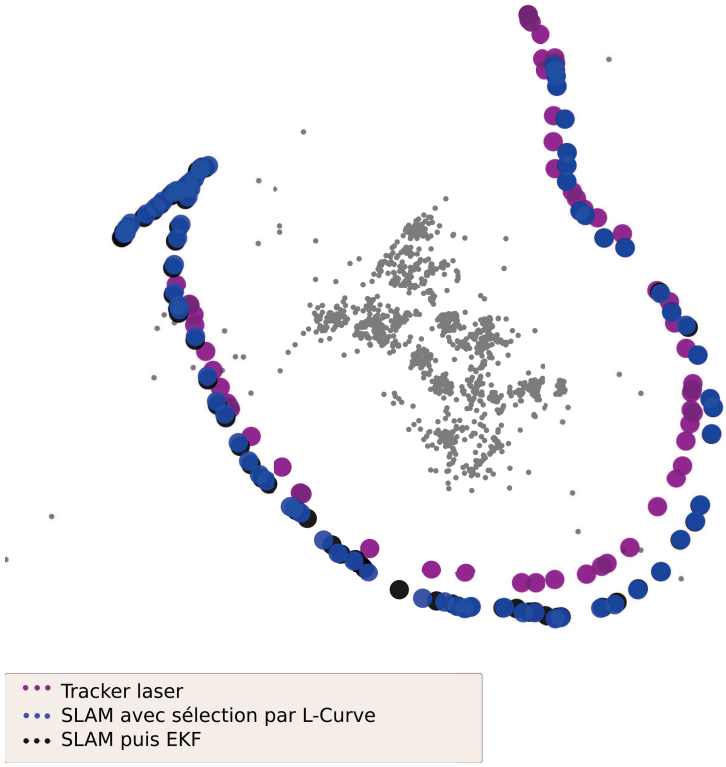


FIGURE 5.19 – Localisations obtenues par les différents SLAM sur la séquence `Gyroviz`. Vue de dessus avec un léger décalage vers le bas.

	médiane *(m)*	**moyenne** *(m)*	**écart-type** *(m)*	**max** *(m)*	**min** *(m)*
SLAM monoculaire	0.18434	0.16002	0.089601	0.28885	0.0046609
SLAM + CV_{loo}	0.12788	**0.11596**	0.06259	0.1943	0.0033829
SLAM + $L-Curve$	0.15492	0.13539	0.078179	0.25124	0.0049847
SLAM + LTN	0.16093	0.14191	0.080251	0.25954	0.0049846
SLAM + EKF	0.18218	0.15762	0.087776	0.28488	0.0047751

TABLE 5.9 – Statistiques sur les trajectoires reconstruites sur la séquence `Gyroviz`.

d'orientation issues du système de navigation. Avec la technique de sélection par validation croisée, nous réduisons l'erreur 3D moyenne d'environ 4.5 cm, soit une amélioration de 38% par rapport au SLAM monoculaire non contraint. Les deux autres techniques de sélection par critère de compromis réduisent aussi mais en moindre mesure l'erreur moyenne de localisation globale.

Notons de plus que la méthode de fusion par un filtre de Kalman *a posteriori* n'apporte pas sur cette séquence un accroissement important de la qualité (environ 3 mm). Il est possible que ce résultat proviennent du fait que nous intégrons dans le filtrage les poses non clés de la caméra. Or, ces poses ne sont pas optimisées lors des ajustement de faisceaux locaux du SLAM et la précision en est réduite. Nous pensons de plus, qu'avec l'ajout de l'information de direction de la gravité, la qualité de la localisation par filtre de Kalman *a posteriori* pourrait être supérieure à

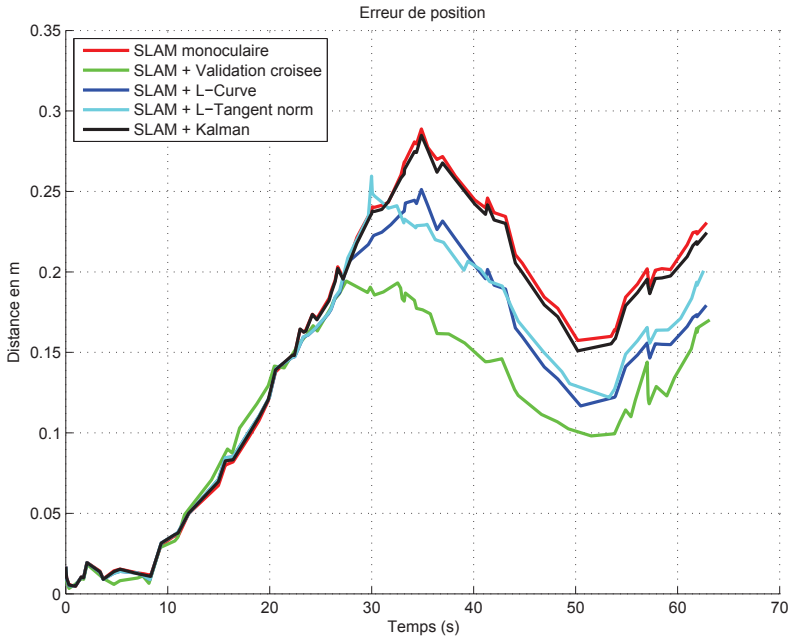


FIGURE 5.20 – Comparaison des méthodes de fusion des orientations sur la séquence `Gyroviz`.

celle que nous observons ici.

Comparaison avec des coefficients de pondération constants. L'objectif ici est de comparer les méthodes de sélection dynamique par rapport à une sélection empirique statique, où les poids sont considérés constants pour toutes les contraintes. La figure 5.21 représente l'erreur moyenne de localisation lorsque l'on fixe la valeur des coefficients de pondération. Cette valeur à été intégrée lors des optimisations de la pose et dans les ajustement de faisceaux locaux, ceci afin que la triangulation puisse bénéficier de l'estimation contrainte de la pose. Nous présentons sur cette figure quelques valeurs de poids allant de 10^{-2} à 20 car au delà, l'algorithme de SLAM n'a pu continuer à localiser la caméra.

Nous remarquons une zone de valeurs de pondération qui améliore nettement la localisation du système. Cette zone s'étale entre 0.2 et 10, avec un minimum aux alentours de 2,4. Le fait est qu'avec une valeur de pondération de 2,4, constante pour toutes les contraintes, l'erreur de localisation 3D moyenne (12.39 cm) est supérieure à ce que l'on obtient en utilisant une sélection dynamique par validation croisée, où l'erreur est de 11.59 cm. Cela signifie qu'il est bénéfique pour la qualité de la localisation globale d'employer une méthode de sélection dynamique du coefficient de pondération. Si l'on compare cette valeur de pondération constante de 2,4, pour laquelle l'erreur de positionnement par rapport à la vérité terrain est minimale, avec la médiane (4.97) des poids sélectionnés tout au long de la séquence par la validation croisée (figure 5.8 et tableau 5.2 page 158), on remarque que ces valeurs sont proches mais non égales. Les médianes des deux autres méthodes de sélection sont quant-à-elles assez faibles par rapport à la valeur de 2,4. Rappelons que le processus de localisation par SLAM est un algorithme incrémental et, par conséquent, une erreur effectuée à un instant donné aura des conséquences importantes sur la

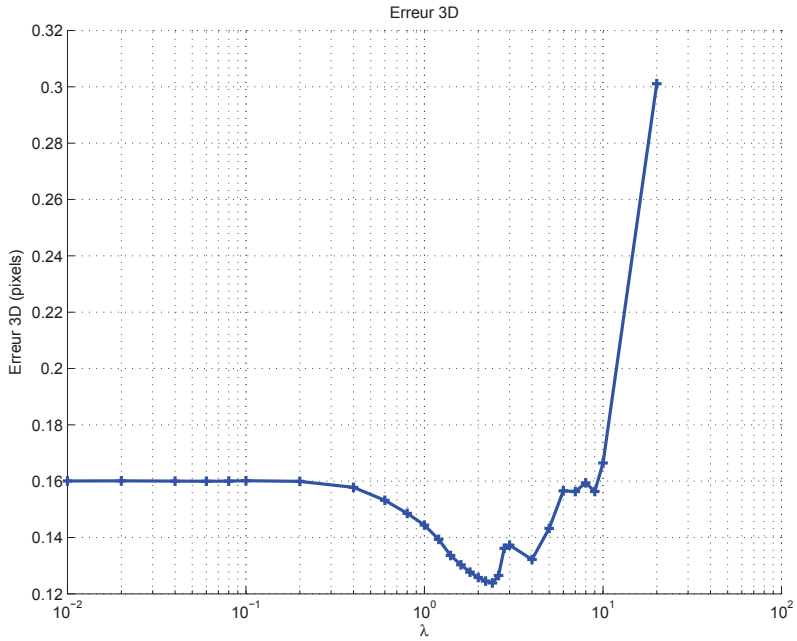


FIGURE 5.21 – Moyennes des erreurs de localisation globale pour différentes valeur fixes de pondérations λ_i.

localisation finale du système. Les résultats montrent que la correction de ces erreurs locales, même avec un faible coefficient de pondération choisi dynamiquement améliore la précision de la localisation finale.

5.4.3.2 Localisation avec une centrale inertielle

La seconde expérimentation de localisation d'une personne en mouvement (libre), utilise la centrale inertielle XSENS.

Présentation. La séquence présentée a été enregistrée dans les couloirs du CEA de Saclay. Un opérateur muni du dispositif composé de la caméra basse résolution AVT GUPPY et de la centrale inertielle XSENS MTx, dont une photographique est proposée dans la figure 5.22, s'est déplacé dans 2 étages d'un bâtiment du CEA. L'opérateur s'est déplacé à vitesse normale (marche) le long du couloir sur une distance d'environ 20 mètres, puis a emprunté un escalier situé sur la gauche afin d'aller à l'étage du dessus, et a finalement marché dans un second couloir sur une distance d'environ 15 mètres. La longueur totale parcourue est environ 40 mètres, sur une durée de 1,5 minutes. Quelques informations sur la localisation et la reconstruction 3D par un SLAM monoculaire sont données dans le tableau 5.10 et trois images de la séquence `Couloir` sont présentées dans la figure 5.23.

Les mesures de la centrale inertielle XSENS-MTx ont été utilisées afin de contraindre les rotations inter-caméras clés du SLAM visuel. La contrainte est $\varepsilon_i^{x,dR}$.

FIGURE 5.22 – Photographie du dispositif de localisation ici placé sur un véhicule. Ce dispositif est composé de la caméra basse résolution AVT GUPPY et de la centrale inertielle XSENS-MTx.

(a) (b) (c)

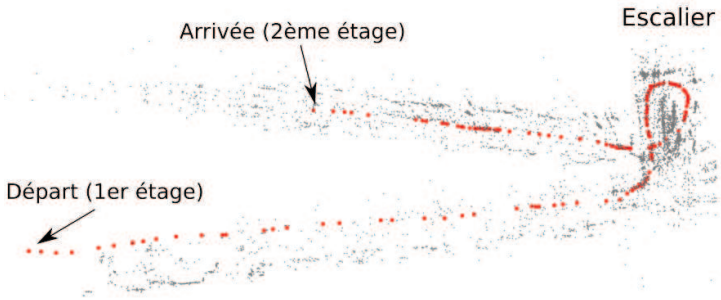
FIGURE 5.23 – Images issues de la séquence `Couloir`.

	Longueur *(m)*	**Nombre d'images (images clés)**	**Nombre de points 3D**
`Couloir`	40	2250 (125)	5221

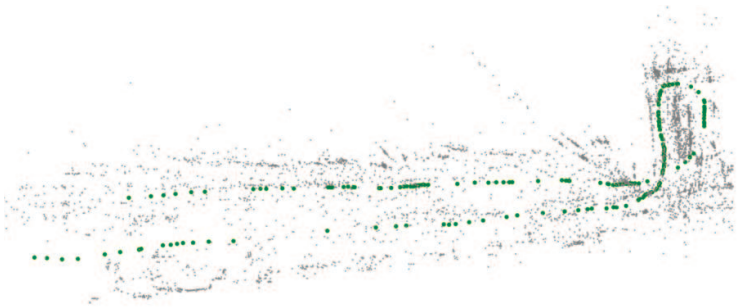
TABLE 5.10 – Statistiques de la localisation par SLAM sur la séquence `Couloir`.

Résultats. Puisque nous n'avons pas de vérité terrain sur cette séquence, nous ne connaissons pas précisément la qualité des localisations produites par les SLAM contraints et non contraint. Nous proposons ici une analyse qualitative et visuelle des résultats, qui permet toutefois de montrer l'intérêt dans des domaines d'utilisation variés.

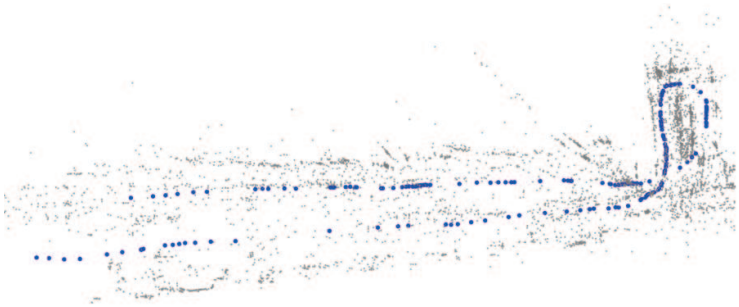
La figure 5.24 présente les trajectoires et les scènes reconstruites avec les différents SLAM non contraint et contraints, en vue de dessus légèrement décalée vers le bas afin de pouvoir différencier les deux étages de la reconstruction. On peut remarquer dans la figure 5.24(a) que le SLAM monoculaire (non contraint) présente une dérive en orientation puisque les deux couloirs des deux étages sont parallèles. Dans les trois reconstructions obtenues avec un SLAM par ajustement de faisceaux local contraint avec les trois méthodes de sélection (figures 5.24(b), 5.24(c) et 5.24(d)), on peut remarquer que la dérive en orientation est réduite, mais non totalement supprimée puisqu'il reste un léger décalage d'angle entre les deux parties de la séquence.



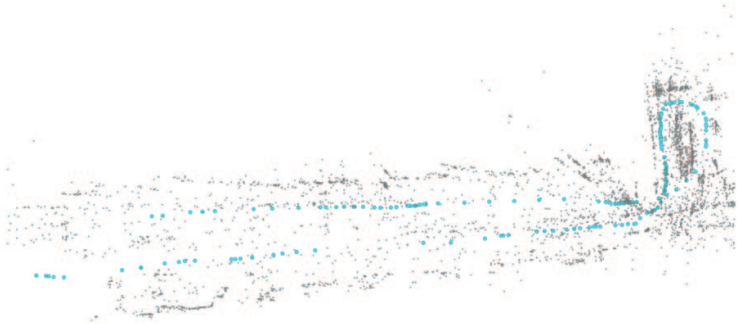
(a) Trajectoire par SLAM monoculaire



(b) Trajectoire par SLAM contraint avec une sélection par validation croisée



(c) Trajectoire par SLAM contraint avec une sélection par L-Curve



(d) Trajectoire par SLAM contraint avec une sélection par *L-Tangent Norm*

FIGURE 5.24 – Trajectoires et reconstructions de la scène pour la séquence `Couloir`. Vues de dessus avec un léger décalage vers le bas.

5.5 Discussion et conclusion

Nous avons présenté dans ce chapitre une méthode permettant d'ajouter des informations de mouvement de la caméra dans un ajustement de faisceaux contraint. Nous nous sommes intéressé plus particulièrement à employer cette technique dans le cadre d'un SLAM monoculaire doté d'un second capteur informant soit de l'orientation de la caméra (par un capteur inertiel ou un système de navigation), soit informant de l'échelle métrique de la trajectoire à l'aide d'un odomètre. Les résultats obtenus sur deux applications très différentes montrent que notre technique améliore la qualité de la localisation du système, en diminuant les dérives observées avec un SLAM monoculaire classique.

Performance de l'approche proposée. Nous avons pu observer dans les expérimentations présentées que le SLAM visuel par ajustement de faisceaux contraint par des informations provenant d'un odomètre ou d'une centrale inertielle, pouvait réduire les dérives observées avec les algorithmes de vision non contraint. Les expérimentations tendent à montrer que la méthode de sélection dynamique par validation croisée est la méthode la plus adaptée dans ce contexte, que ce soit pour fusionner avec un odomètre ou avec un système de navigation. Les méthodes de sélection par critère de compromis améliorent aussi la qualité globale de la localisation, mais en moindre mesure. Ces critères sont souvent instables, à cause de l'estimation des dérivées, mais ils ont l'avantage d'être plus rapide à estimer que le critère de validation croisée.

Limites de la méthode actuelle. Le SLAM multi-capteur proposé dans cette thèse fonctionne en temps réel et peut être utilisé avec différents capteurs, comme notamment les centrales inertielles ou les odomètres. Notons cependant que l'approche incrémentale proposée, a été conçue pour corriger localement la trajectoire du SLAM, à l'aide de capteurs délivrant fréquemment leurs observations (haute fréquence). La méthode actuelle n'est clairement pas adaptée pour des capteurs à basses fréquences comme le GPS pour plusieurs raisons. Premièrement, nous n'appliquons pas de correction globale : nous ne remettons jamais en cause l'historique de la trajectoire (sauf par l'ajustement de faisceaux local, mais cela reste très local). De plus, la sélection des coefficients de pondération associés aux contraintes est effectuée de façon locale (pour être temps réel), sur un critère basé uniquement sur la dernière pose du système. Si la distance entre la pose estimée par resection et la pose mesurée par le GPS par exemple est trop importante, alors les poids sélectionnés seront *a priori* nuls, ou très faibles puisqu'ils dégraderont rapidement la localisation.

La méthode proposée améliore la localisation car les corrections apportées par les contraintes sont souvent légères et régulières. Pour cela, il serait peut être intéressant de sélectionner les coefficients non seulement en optimisant la pose courante, mais en ajoutant les points 3D de la scène observés dans l'image (ou une partie) à l'optimisation et les quelques dernières poses de la trajectoire. Pour des raisons de temps de calcul, nous avons fait l'hypothèse que les points 3D sur lesquels la pose courante était optimisée lors de la sélection des poids, était précisément reconstruits.

Par ailleurs, les poids des contraintes ne sont estimés qu'une seule et unique fois, puisque l'on ne considère que les ratios des deux objectifs (erreurs de reprojection et contrainte) de chaque pose varient peu au cours des optimisations.

Conclusion

Travaux réalisés

Les travaux présentés dans ce manuscrit s'intéressent à améliorer l'algorithme d'ajustement de faisceaux, d'une part en accélérant le processus de minimisation par une technique de recherche linéaire algébrique, et d'autre part en intégrant des informations issues de capteurs de mouvement.

Un algorithme d'ajustement de faisceaux rapide. Nous avons proposé un algorithme d'ajustement de faisceaux accéléré qui intègre une nouvelle technique de recherche linéaire non itérative. Cette technique, la recherche linéaire algébrique, simplifie l'erreur de reprojection de l'ajustement de faisceaux en utilisant une distance algébrique, et rend possible le calcul analytique d'une amplitude (ou de deux amplitudes pour la variante *Two way-ALS*) de déplacement efficace, par la résolution d'un polynôme de degré 3 (*G-ALS*) ou 5 (*Two way-ALS*). Les algorithmes proposés sont efficaces, rapides et ils peuvent être employés dans différents contextes : *batch Structure-from-Motion*, calcul de pose, triangulation ou tout autre problème nécessitant la minimisation de l'erreur de reprojection. Les expérimentations conduites sur des données simulées et réelles sur une application de reconstruction 3D hors ligne par *batch Structure-from-Motion*, montrent que cette technique est plus rapide que les méthodes de l'état de l'art et permet d'accélérer la convergence des ajustements de faisceaux calibrés et non calibrés. Notons de plus que les gains en itérations et en temps de calcul sont d'autant plus importants que la solution initiale du problème est peu précise.

Un algorithme de SLAM multi-capteur par ajustement de faisceaux contraint. Nous avons réalisé un SLAM multi-capteur temps réel basé sur les algorithmes de vision (*Structure-from-Motion* incrémental) et qui intègre les informations de mouvement provenant d'un second capteur dans les ajustements de faisceaux locaux, sous la forme de contraintes pondérées. La sélection des coefficients de pondération est effectuée dynamiquement par deux types d'approches : par validation croisée (*leave-one-out*) et par deux critères de compromis (L-Curve et LTN). Nous avons observé que ces deux types d'approches sélectionnent des coefficients assez différents, à l'image de leur critère : la validation croisée privilégie les coefficients qui améliorent la prédictivité du modèle, alors que les critères de compromis recherchent le coefficient maximisant la courbure du compromis (L-Curve), ou minimisant la norme de la tangente (LTN). Les expérimentations montrent cependant que les meilleurs résultats de localisation sont obtenus par la méthode de sélection par validation croisée. L'origine du capteur varie suivant l'application ciblée. Nous avons validé le SLAM par ajustement de faisceaux contraint sur deux applications : la localisation d'un véhicule à l'aide d'un odomètre, et la localisation d'un système embarqué sur un individu avec les orientations d'une centrale inertielle ou d'un système de navigation à guidage inertiel. Nos expérimentations menées sur des séquences réelles montrent que

cette technique d'ajustement de faisceaux contraint réduit les dérives observées avec un SLAM monoculaire classique, ce qui contribue à l'amélioration de la qualité de la localisation finale du système.

Perspectives

La recherche linéaire algébrique. Un travail intéressant et non effectué serait de valider la méthode de recherche linéaire algébrique sur des problèmes à très grandes dimensions (reconstruction d'une ville par exemple), où le nombre de variables optimisées simultanément est très important, et pour laquelle la solution initiale du problème est généralement éloignée de l'optimum.

De plus, nous avons mentionné dans ce manuscrit que puisque la recherche linéaire algébrique emploie une approximation de l'erreur de reprojection différente de l'approximation de TAYLOR effectuée par l'algorithme de LEVENBERG-MARQUARDT, cette technique pourrait être utilisée afin d'éviter les minima locaux de la fonction de coût. Une étude sur la capacité de la recherche linéaire algébrique à détecter et éviter ces minima locaux nous semble utile.

Le SLAM multi-capteur par ajustement de faisceaux contraint. La méthode actuellement proposée intègre des informations de déplacement (odomètre, centrale inertielle) par des contraintes inter-caméras clés. Il serait intéressant d'améliorer la méthode afin d'intégrer des données de positionnement absolu provenant d'un GPS ou d'un algorithme de reconnaissance de lieu par exemple (correction globale).

En outre, nous avons proposé d'intégrer les observations d'un seul capteur supplémentaire dans un SLAM visuel et il serait utile d'étendre la méthode à plusieurs capteurs. L'ajustement de faisceaux contraint peut être simplement étendu à plusieurs capteurs en sommant l'ensemble des contraintes, mais le choix des coefficients de pondération peut poser des difficultés, surtout si l'on désire employer cette méthode en temps réel. Notons à ce propos qu'il existe une généralisation en $n > 2$ dimensions de certains critères comme la L-Curve par exemple ([Belge 2002]).

Enfin, d'autres applications pourrait bénéficier de ce processus d'ajustement de faisceaux contraint avec la sélection automatique des poids, par exemple l'autocalibrage de caméra, lors d'un zoom par exemple, ou en intégrant des contraintes (*a priori*) sur la structure de la scène (recalage sur un modèle CAO par exemple).

Bibliographie

[Agarwal 2009] S. Agarwal, N. Snavely, I. Simon, S.M. Seitz et R. Szeliski. *Building Rome in a day*. In ICCV, pages 72–79, 2009. 11, 12, 207

[Agarwal 2010] Sameer Agarwal, Noah Snavely, Steven M. Seitz et Richard Szeliski. *Bundle Adjustment in the Large*. In ECCV, 2010. 98

[Agrawal 2006a] M. Agrawal. *A Lie Algebraic Approach for Consistent Pose Registration for General Euclidean Motion*. pages 1891 –1897, oct. 2006. 138

[Agrawal 2006b] M. Agrawal et K. Konolige. *Real-time Localization in Outdoor Environments using Stereo Vision and Inexpensive GPS*. In International Conference on Pattern Recognition, August 2006. 13, 136, 138

[Al-Baali 1986] M. Al-Baali et R. Fletcher. *An efficient line search for nonlinear least squares*. J. Optim. Theory Appl., vol. 48, no. 3, pages 359–377, 1986. 47

[Angeli 2008] A. Angeli, D. Filliat, S. Doncieux et J.-A. Meyer. *Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words*. Robotics, IEEE Transactions on, vol. 24, no. 5, pages 1027 –1037, oct. 2008. 13, 136

[Armesto 2007] Leopoldo Armesto, Josep Tornero et Markus Vincze. *Fast Ego-motion Estimation with Multi-rate Fusion of Inertial and Vision*. International Journal of Robotics Research, vol. 26, pages 577–589, 2007. 13, 136, 137

[Bartoli 2002] Adrien Bartoli. *A Unified Framework for Quasi-Linear Bundle Adjustment*. Pattern Recognition, International Conference on, vol. 2, page 20560, 2002. 85, 102

[Bartoli 2004] Adrien Bartoli et Peter Sturm. *Nonlinear Estimation of the Fundamental Matrix with Minimal Parameters*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 3, pages 426–432, 2004. 97

[Bartoli 2008] Adrien Bartoli. *Maximizing the Predictivity of Smooth Deformable Image Warps through Cross-Validation*. J. Math. Imaging Vis., vol. 31, no. 2-3, pages 133–145, 2008. 140, 159

[Bay 2006] Herbert Bay, Tinne Tuytelaars et Luc Van Gool. *SURF : Speeded Up Robust Features*. In Ales Leonardis, Horst Bischof et Axel Pinz, editeurs, ECCV, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin Heidelberg, 2006. 65

[Bazán 2008] F. S. Bazán. *Fixed-point iterations in determining the Tikhonov regularization parameter*. Inverse Problems, vol. 24, pages 1–15, 2008. 59, 60

[Bazán 2009] Fermín S Viloche Bazán et Juliano B Francisco. *An improved fixed-point algorithm for determining a Tikhonov regularization parameter*. Inverse Problems, vol. 25, 2009. 59, 60, 140

[Beaton 1974] Albert E. Beaton et John W. Tukey. *The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data*. Technometrics, vol. 16, no. 2, pages 147–185, 1974. 55

[Belge 2002] Murat Belge, Misha E Kilmer et Eric L Miller. *Efficient determination of multiple regularization parameters in a generalized L-curve framework*. Inverse Problems, vol. 18, no. 4, page 1161, 2002. 186

[Bishop 2007] Christopher M. Bishop. Pattern recognition and machine learning (information science and statistics). Springer, 1 édition, 2007. 149

[Björk 1996] Å. Björk. *Numerical Methods for Least Squares Problems*. SIAM, 1996. 38

[Bouguet 2008] J. Y. Bouguet. *Camera calibration toolbox for Matlab*, 2008. 26

[Bradski 2000] G. Bradski. *The OpenCV Library*. Dr. Dobb's Journal of Software Tools, 2000. 26

[Broyden 1965] C. G. Broyden. A class of methods for solving nonlinear simultaneous equations, volume 19. American Mathematical Society, 1965. 45

[Brunet 2008] F. Brunet, A. Bartoli, R. Malgouyres et N. Navab. *L-Tangent Norm : A Low Computational Cost Criterion for Choosing Regularization Weights and its Use for Range Surface Reconstruction*. In 3D Data Processing, Visualization and Transmission (3DPVT), 2008. 59, 61, 140

[Buchanan 2005] A.M. Buchanan et A.W. Fitzgibbon. *Damped Newton algorithms for matrix factorization with missing data*. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 2, pages 316 – 322 vol. 2, 20-25 2005. 97

[Byröd 2009] Martin Byröd et Kalle Åström. *Bundle Adjustment using Conjugate Gradients with Multiscale Preconditioning*. In BMVC, 2009. 98

[Carasso 1982] A. S. Carasso. *Determining Surface Temperatures from Interior Observations*. SIAM J. Appl. Math., vol. 42, pages 558–574, 1982. 59

[Chen 2009] Chao-I Chen, Dusty Sargent, Chang-Ming Tsai, Yuan-Fang Wang et Dan Koppel. *Uniscale Multi-view Registration Using Double Dog-Leg Method*. Progress in biomedical optics and imaging, vol. 10, 2009. 97

[Chenavier 1992] F Chenavier et J L Crowley. *Position estimation for a mobile robot using vision and odometry*. In IEEE International Conference on Robotics and Automation, 1992. 13, 136

[Chojnacki 2003] W. Chojnacki et M.J. Brooks. *Revisiting Hartley's normalized eight-point algorithm*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 9, pages 1172–1177, Sept. 2003. 80, 104

[Chui 2003] Haili Chui et Anand Rangarajan. *A new point matching algorithm for non-rigid registration*. Computer Vision and Image Understanding, vol. 89, no. 2-3, pages 114–141, 2003. 140

[Civera 2009] Javier Civera, Oscar G. Grasa, Andrew J. Davison et J. M. M. Montiel. *1-Point RANSAC for EKF-Based Structure from Motion*. In International Conference on Intelligent Robots and Systems, 2009. 137

[Craciun 2008] D. Craciun, N. Paparoditis et F. Schmitt. *Automatic Pyramidal Intensity-based Laser Scan Matcher for 3D Modeling of Large Scale Unstructured Environments*. In Canadian Conference on Computer and Robot Vision, May 2008. 136

[Davison 2003] Andrew J. Davison. *Real-Time Simultaneous Localisation and Mapping with a Single Camera*. In International Conference on Computer Vision, page 1403, Washington, DC, USA, 2003. IEEE Computer Society. 88, 136, 137

[Dellaert 2010] Frank Dellaert, Justin Carlson, Viorela Ila, Kai Ni et Charles E. Thorpe. *Subgraph-preconditioned Conjugate Gradients for Large Scale SLAM*. In IROS, 2010. 98

[Dennis 1996] J. E. Dennis Jr. et Robert B. Schnabel. Numerical methods for unconstrained optimization and nonlinear equations (classics in applied mathematics, 16). Soc for Industrial & Applied Math, 1996. 97

[Eade 2006] Ethan Eade et Tom Drummond. *Scalable monocular SLAM*. In Computer Vision and Pattern Recognition, pages 469–476. IEEE Computer Society, 2006. 137

[Eade 2008] Ethan Eade et Tom Drummond. *Unified loop closing and recovery for real time monocular slam*. In BMVC, 2008. 13, 136

[Engels 2006] Chris Engels, Henrik Stewénius et David Nistér. *Bundle adjustment rules*. In In Photogrammetric Computer Vision, 2006. 86, 88, 100

[Eudes 2009] A. Eudes et M. Lhuillier. *Error Propagations for Local Bundle Adjustment*. In Computer Vision and Pattern Recognition, Miami, Florida, June 2009. 139

[Eudes 2010a] A. Eudes, Maxime Lhuillier, Sylvie Naudet-Collette et Michel Dhome. *Fast Odometry Integration in Local Bundle Adjustment-based Visual SLAM*. In ICPR, 2010. 13, 136

[Eudes 2010b] A. Eudes, Sylvie Naudet-Collette, Maxime Lhuillier et Michel Dhome. *Weighted Local Bundle Adjustment and Application to Odometry and Visual SLAM Fusion*. In BMVC2010, 2010. 139, 140

[Farenzena 2008] Michela Farenzena, Adrien Bartoli et Youcef Mezouar. *Efficient Camera Smoothing in Sequential Structure-from-Motion Using Approximate Cross-Validation*. In European Conference on Computer Vision, pages 196–209, 2008. 57, 140, 141, 142, 143, 159

[Farenzena 2009] Michela Farenzena, Andrea Fusiello et Riccardo Gherardi. *Structure-and-Motion Pipeline on a Hierarchical Cluster Tree*. In 3DIM, 2009. 86

[Fasel 2008] Ian Fasel et Paul Ruvolo. *Optimization on a Budget : A Reinforcement Learning Approach*. In Neural Information Processing Systems (NIPS), 2008. 51

[Faugeras 1993] O.D. Faugeras. Three-dimensional computer vision : A geometric viewpoint. MIT Press, 1993. 70

[Finsterwalder 1937] S. Finsterwalder et W. Scheufele. *Das Rückwärtsein-schneiden im Raum*, 1937. 68

[Fischler 1981] Martin A. Fischler et Robert C. Bolles. *Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography*. Commun. ACM, vol. 24, no. 6, pages 381–395, 1981. 55

[Fitzgibbon 1998] A. W. Fitzgibbon et A. Zisserman. *Automatic Camera Recovery for Closed or Open Image Sequences*. In Proceedings of the European Conference on Computer Vision, pages 311–326. Springer-Verlag, Juin 1998. 86

[Forsyth 2002] David A. Forsyth et Jean Ponce. Computer vision : A modern approach. Prentice Hall, us ed édition, August 2002. 22

[Frahm 2010] J.M. Frahm, P. Fite Georgel, D. Gallup, T. Johnson, R. Raguram, C.C. Wu, Y.H. Jen, E. Dunn, B. Clipp, S. Lazebnik et M. Pollefeys. *Building Rome on a Cloudless Day*. In ECCV, pages IV : 368–381, 2010. 11

[Frandsen 1999] P.E. Frandsen, K. Jonasson, H.B. Nielsen et O. Tingleff. Unconstrained optimization. 1999. 43, 45, 46, 47, 120, 121, 123, 130

[Gemeiner 2007] Peter Gemeiner, Peter Einramhof et Markus Vincze. *Simultaneous Motion and Structure Estimation by Fusion of Inertial and Vision Data*. Int. J. Rob. Res., vol. 26, no. 6, pages 591–605, 2007. 13, 136, 137

[Golub 1979] G. H. Golub, M. Heath et G. Wahba. *Generalized cross-validation as a method for choosing a good ridge parameter*. Technometrics, vol. 21, page 215–223, 1979. 57

[Granshaw 1980] S. I. Granshaw. *Bundle adjustment methods in engineering photgrammetry*. The Photogrammetric Record, vol. 10, pages 1477–9730, 1980. 98

[Grünert 1841] J. Grünert. *Das Pothenotische Problem in erweiterter Gestalt nebst über seine Anwendungen in der Geodäsie*. Grunerts Archiv für Mathematik und Physik, vol. 1, page 238–248, 1841. 67, 68

[Guilbert 2006] Nicolas Guilbert, Adrien Bartoli et Anders Heyden. *Affine Approximation for Direct Batch Recovery of Euclidian Structure and Motion from Sparse Data*. International Journal of Computer Vision, vol. 69, pages 317–333, 2006. 10.1007/s11263-006-8113-4. 76

[Gurdjos 2003] P. Gurdjos et P. Sturm. *Methods and geometry for plane-based self-calibration*. In Computer Vision and Pattern Recognition, volume 1, pages I–491 – I–496 vol.1, jun. 2003. 26

[Hager 2006a] W. Hager et H. Zhang. *Algorithm 851 : CG DESCENT, a conjugate gradient method with guaranteed descent*. ACM Trans. Math. Softw., vol. 32, no. 1, pages 113–137, 2006. 47

[Hager 2006b] William W. Hager et Hongchao Zhang. *A survey of nonlinear conjugate gradient methods*. Pacific Journal of optimization, vol. Volume 2, pages 35–58, 2006. 47

[Hansen 2001] P. C. Hansen. *The L-Curve and its Use in the Numerical Treatment of Inverse Problems*. In Computational Inverse Problems in Electrocardiology, numéro 5 de Advances in Computational Bioengineering, pages 119–142. WIT Press, Southampton, 2001. 59, 60, 140

[Haralick 1994] Robert M. Haralick, Chung-Nan Lee, Karsten Ottenberg et Michael Nölle. *Review and analysis of solutions of the three point perspective pose estimation problem*. International Journal of Computer Vision, vol. 13, no. 3, pages 331–356, 1994. 67, 68

[Harris 1988] C. Harris et M. Stephens. *A Combined Corner and Edge Detection*. In The Fourth Alvey Vision Conference, pages 147–151, 1988. 64

[Hartley 1995] Richard. Hartley. *In defence of the 8-point algorithm*. In International Conference on Computer Vision, page 1064, Washington, DC, USA, 1995. IEEE Computer Society. 70, 72, 97

[Hartley 1997] Richard I. Hartley et Peter Sturm. *Triangulation*. In Computer Vision and Image Understanding, volume 68, pages 146–157, New York, NY, USA, 1997. Elsevier Science Inc. 66

[Hartley 1998] Richard. Hartley. *Minimizing Algebraic Error*. In International Conference on Computer Vision, page 469, Washington, DC, USA, 1998. IEEE Computer Society. 79, 104

[Hartley 1999] Richard. Hartley. *Camera Calibration and the Search for Infinity*. In ICCV, pages 510–517, 1999. 86, 88, 102

[Hartley 2003a] Richard Hartley et Frederik Schaffalitzky. *PowerFactorization : 3D reconstruction with missing or uncertain data*. In Australia-Japan Advanced Workshop on Computer Vision, 2003. 74, 76, 85, 111

[Hartley 2003b] Richard. Hartley et Andrew. Zisserman. Multiple view geometry in computer vision. Cambridge University Press, Cambridge, UK, second édition, 2003. 34, 52, 55, 70, 73, 74, 80, 83, 87, 88, 97, 120

[Hestenes 1952] Magnus R. Hestenes et Eduard Stiefel. *Methods of Conjugate Gradients for Solving Linear Systems*. Journal of Research of the National Bureau of Standards, vol. 49, no. 6, pages 409–436, December 1952. 43

[Heyden 1997] Anders Heyden. *Projective Structure and Motion from Image Sequences using Subspace Methods*. Scandinavian Conferences on Image Analysis, vol. 1, pages 963–968, 1997. 74, 75

[Higham 2001] Nicholas J. Higham et Hyun-Min Kim. *Solving a Quadratic Matrix Equation by Newton's Method with Exact Line Searches*. SIAM J. Matrix Anal. Appl., vol. 23, no. 2, pages 303–316, 2001. 45

[Hol 2006] J.D. Hol, T.B. Schon, F. Gustafsson et P.J. Slycke. *Sensor Fusion for Augmented Reality*. In Information Fusion, pages 1–6, July 2006. 13, 136

[Hol 2007] Jeroen Hol, Thomas Schön, H. Luinge, P. Slycke et Fredrik Gustafsson. *Robust real-time tracking by fusing measurements from inertial and vision sensors*. Journal of

Real-Time Image Processing, vol. 2, no. 2, pages 149–160, Novembre 2007. 13, 136, 137

[Howard 2008] A. Howard. *Real-time stereo visual odometry for autonomous ground vehicles*. Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, pages 3946 –3952, sep. 2008. 136

[Huber 1981] P. Huber. Robust statistics. Wiley, 1981. 55

[Hung 2006] Y.S. Hung et W.K. Tang. *Projective Reconstruction from Multiple Views with Minimization of 2D Reprojection Error*. International Journal of Computer Vision, vol. 66, page 305–317, 2006. 97

[Huster 2003] Andreas Huster. *Relative position sensing by fusing monocular vision and inertial rate sensors*. PhD thesis, Stanford University Stanford, Stanford, CA, USA, 2003. 13, 136, 137

[Jacobs 1997] David Jacobs. *Linear Fitting with Missing Data : Applications to Structure-from-Motion and to Characterizing Intensity Images*. Computer Vision and Pattern Recognition, vol. 0, page 206, 1997. 76

[Jeong 2010] Yekeun Jeong, David Nister, Drew Steedly, Richard Szeliski et InSo Kweon. *Pushing the Envelope of Modern Methods for Bundle Adjustment*. In Conference on Computer Vision and Pattern Recognition, 2010. 98, 99

[Jung 2001] S.-H. Jung et C.J. Taylor. *Camera trajectory estimation using inertial sensor measurements and structure from motion results*. In Computer Vision and Pattern Recognition, volume 2, pages II–732 – II–737 vol.2, 2001. 138

[Kaess 2009] Michael Kaess et Frank Dellaert. *Multi-Camera Visual SLAM*. Computer Vision and Image Understanding, 2009. 136

[Kahl 2001] F. Kahl et A. Heyden. *Euclidean reconstruction and auto-calibration from continuous motion*. In ICCV, volume 2, pages 572 –577 vol.2, 2001. 142, 143

[Kanade 1992] Takeo Kanade et Carlo Tomasi. *Shape and motion from image streams under orthography : a factorization method*. International Journal of Computer Vision, vol. 9, pages 137–154, 1992. 74, 76

[Klein 2007] Georg Klein et David Murray. *Parallel Tracking and Mapping for Small AR Workspaces*. In International Symposium on Mixed and Augmented Reality, Nara, Japan, November 2007. 11, 12, 86, 88, 97, 136, 207

[Konolige 2007] Kurt Konolige, Motilal Agrawal et Joan Solà. *Large scale visual odometry for rough terrain*. In International Symposium on Robotics Research, 2007. 138, 177

[Konolige 2010] Kurt Konolige. *Sparse Sparse Bundle Adjustment*. In British Machine Vision Conference, 2010. 99

[Krawczy-Stando 2007] M. Krawczy-Stando D.and Rudnicki. *Regularization parameter selection in discrete-ill posed problems - the use of the U-curve*. Int. J. Appl. Math. Comput. Sci., vol. 17, pages 157–64, 2007. 59

[Krawczyk-StańDo 2007] Dorota Krawczyk-StańDo et Marek Rudnicki. *Regularization Parameter Selection in Discrete Ill-Posed Problems-The Use of the U-Curve*. Int. J. Appl. Math. Comput. Sci., vol. 17, no. 2, pages 157–164, 2007. 140

[Lavest 1998] Jean-Marc Lavest, Marc Viala et Michel Dhome. *Do We Really Need an Accurate Calibration Pattern to Achieve a Reliable Camera Calibration ?* In ECCV '98 : Proceedings of the 5th European Conference on Computer Vision-Volume I, pages 158–174, London, UK, 1998. Springer-Verlag. 26

[Lepetit 2009] Vincent Lepetit, Francesc Moreno-Noguer et Pascal Fua. *EPnP : An Accurate O(n) Solution to the PnP Problem*. International Journal of Computer Vision, vol. 81, no. 2, pages 155–166, 2009. 68

[Levenberg 1944] K. Levenberg. *A Method for the Solution of Certain Non-linear Problems in Least Squares.* Quarterly of Applied Mathematics, 1944. 51

[Liu 1989] D.C. Liu et J. Nocedal. *On the limited memory BFGS method for large scale optimization*. Math. Program., vol. 45, no. 3, pages 503–528, 1989. 45, 47

[Liu 2003] Yufeng Liu et Sebastian Thrun. *Results for outdoor-SLAM using sparse extended information filters*. In in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA, pages 1227–1233, 2003. 137

[Liu 2008] Shigang Liu, Jiancheng Sun et Jianwu Dang. *A Linear Resection-Intersection Bundle Adjustment Method*. Information Technology Journal, vol. 7, pages 220–223, 2008. 85

[Lobo 2007] Jorge Lobo et Jorge Dias. *Relative Pose Calibration Between Visual and Inertial Sensors*. Int. J. Rob. Res., vol. 26, no. 6, pages 561–575, 2007. 167

[Longuet-Higgins 1981] H. C. Longuet-Higgins. *A computer algorithm for reconstructing a scene from two projections*. Nature, vol. 293, page 133–135, 1981. 70, 71, 72

[Lothe 2009] Pierre Lothe, Steve Bourgeois, Fabien Dekeyser, Eric Royer et Michel Dhome. *Towards Geographical Referencing of Monocular SLAM Reconstruction Using 3D City Models : Application to Real-Time Accurate Vision-Based Localization*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09), Miami, Florida, June 2009. 13, 136

[Lothe 2010] Pierre Lothe, Steve Bourgeois, Eric Royer, Michel Dhome et Sylvie Naudet Collette. *Real-Time Vehicle Global Localisation with a Single Camera in Dense Urban Areas : Exploitation of Coarse 3D City Models*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2010), San Francisco, California, June 2010. 136

[Lourakis 2005] M. Lourakis et A. Argyros. *Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment ?* International Conference on Computer Vision, 2005. 50, 97, 120, 122

[Lourakis 2009] M.I. A. Lourakis et A.A. Argyros. *SBA : A Software Package for Generic Sparse Bundle Adjustment*. ACM Trans. Math. Software, vol. 36, no. 1, pages 1–30, 2009. 97, 99

[Lowe 2004] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints*. International Journal of Computer Vision, vol. 60, pages 91–110, 2004. 64, 65

[Luong 1995] Q.-T. Luong et O.D. Faugeras. *The Fundamental matrix : theory, algorithms, and stability analysis*. International Journal of Computer Vision, vol. 17, pages 43–75, 1995. 97

[Malis 2006] Ezio Malis et Eric Marchand. *Experiments with robust estimation techniques in real-time robot vision*. Intelligent Robots and Systems, pages 223 –228, 2006. 56

[Marquardt 1963] D. Marquardt. *An Algorithm for Least-Squares Estimation of Nonlinear Parameters*. SIAM Journal on Applied Mathematics, vol. 11, pages 431, 441, 1963. 51

[Martinec 2005] Daniel Martinec et Tomáš Pajdla. *3D Reconstruction by Fitting Low-rank Matrices with Missing Data*. Computer Vision and Pattern Recognition, vol. 1, pages 198–205, 2005. 74, 75, 76

[Martinec 2008] Daniel Martinec. *Robust Multiview Reconstruction*. PhD thesis, Faculty of Electrical Engineering Czech Technical University in Prague, 2008. 86, 87

[Michot 2009a] Julien Michot, Adrien Bartoli et François Gaspard. *Algebraic Line Search for Bundle Adjustment*. In BMVC, 2009. 14, 95

[Michot 2009b] Julien Michot, Adrien Bartoli et François Gaspard. *Une longueur de pas optimale au sens de l'erreur de reprojection algébrique pour l'ajustement de faisceaux*. In ORASIS, Tregastel, June 2009. Line Search, moindres carrés non-linéaires, ajustement defaisceaux, Structure-from-Motion. 14, 95

[Michot 2009c] Julien Michot, Adrien Bartoli, François Gaspard et Jean-Marc Lavest. *Longueur de pas algébrique pour l'ajustement de faisceaux*. In CORESA, pages 146–151, Toulouse, March 2009. 14, 95

[Michot 2010a] Julien Michot, Adrien Bartoli et François Gaspard. *Bi-Objective Bundle Adjustment with Application to Multi-Sensor SLAM*. In 3DPVT, paris, france, 2010. 14, 135

[Michot 2010b] Julien Michot, Adrien Bartoli et François Gaspard. *Procédé et système pour fusionner des données issues de capteurs d'image et de capteurs de mouvement ou de position*, 2010. num. 1053803, déposé le 17/05/2010. 14, 135

[Mikhail 2001] Edward M. Mikhail, James S. Bethel et J. Chris McGlone. Introduction to modern photogrammetry. WileyMarch, 2001. 26

[Mikolajczyk 2004] Krystian Mikolajczyk et Cordelia Schmid. *Scale & Affine Invariant Interest Point Detectors*. Int. J. Comput. Vision, vol. 60, no. 1, pages 63–86, 2004. 64

[Mirisola 2007] Luiz G. B. Mirisola, Jorge Lobo et Jorge Dias. *3D Map Registration using Vision/Laser and Inertial Sensing*. In ECMR, 2007. 13, 136

[Modersitzki 2004] Jan Modersitzki. Numerical methods for image registration. Oxford University Press,, 2004. 140

[Montemerlo 2002] Michael Montemerlo, Sebastian Thrun, Daphne Koller et Ben Wegbreit. *FastSLAM : A Factored Solution to the Simultaneous Localization and Mapping Problem*. In Artificial Intelligence, pages 1151–1156, 2002. 137

[Montemerlo 2003] Michael Montemerlo, Sebastian Thrun, Daphne Koller et Ben Wegbreit. *FastSLAM 2.0 : An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges*. In In Proc. of the Int. Conf. on Artificial Intelligence (IJCAI, pages 1151–1156, 2003. 137

[Montiel 2006] J. Montiel, J. Civera et A. Davison. *Unified Inverse Depth Parametrization for Monocular SLAM*. In Robotics : Science and Systems, Philadelphia, USA, August 2006. 137

[Moré 1994] J. Moré et D. Thuente. *Line search algorithms with guaranteed sufficient decrease*. ACM Trans. Math. Softw., vol. 20, no. 3, pages 286–307, 1994. 47

[Morris 1998] Daniel Morris et Takeo Kanade. *A Unified Factorization Algorithm for Points, Line Segments and Planes with Uncertainty Models*. International Conference on Computer Vision, pages 696–702, 1998. 85

[Mouragnon 2006] Etienne Mouragnon, Maxime Lhuiller, Michel Dhome, Fabien Dekeyser et Patrick Sayd. *Real Time Localization and 3D Reconstruction*. In Computer Vision and Pattern Recognition, June 2006. 88, 89, 92, 97, 99, 100, 136, 137, 138, 139, 142, 170, 207

[Mouragnon 2007] Étienne Mouragnon. *Reconstruction 3D et localisation simultanée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local*. PhD thesis, Université Blaise Pascal - Clermont-Ferrand II, 2007. 81, 86, 88, 161, 163

[Naikal 2009] N. Naikal, J. Kua, G. Chen et A. Zakhor. *Image augmented laser scan matching for indoor dead reckoning*. In International Conference on Intelligent RObots and Systems (IROS), pages 4134 –4141, oct. 2009. 13, 136

[Newcombe 2010] Richard A. Newcombe et Andrew J. Davison. *Live Dense Reconstruction with a Single Moving Camera*. In CVPR, 2010. 11, 12, 207

[Newman 2006] P. Newman, D. Cole et K. Ho. *Outdoor SLAM using visual appearance and laser ranging*. In ICRA, pages 1180 –1187, may. 2006. 13, 136

[Ni 2007] Kai Ni, D. Steedly et F. Dellaert. *Out-of-Core Bundle Adjustment for Large-Scale 3D Reconstruction*. In Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, pages 1 –8, 14-21 2007. 97, 100

[Ni 2010] Kai Ni et Frank Dellaert. *Multi-Level Submap Based SLAM Using Nested Dissection*. In IROS, 2010. 86

[Nielsen 1999] H.B. Nielsen. *Damping parameter in Marquardt's method*. Rapport technique, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, apr 1999. 51, 120

[Nistér 2000] David Nistér. *Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors*. In European Conference on Computer Vision, volume 1, pages 649–663, 2000. 86

[Nistér 2004a] David Nistér. *An Efficient Solution to the Five-Point Relative Pose Problem*. Pattern Analysis and Machine Intelligence, vol. 26(6), pages 756–770, 2004. 70, 72, 73

[Nistér 2004b] David Nistér. *Untwisting a Projective Reconstruction*. International Journal of Computer Vision, vol. 60, pages 165–183, 2004. 10.1023/B :VISI.0000029667.76852.a1. 86, 88

[Nocedal 1992] J. Nocedal et Y. Yuan. *Combining trust region and line search techniques*. Rapport technique, Advances in Nonlinear Programming, (Kluwer), 1992. 47, 103

[Nocedal 1999] J. Nocedal et S.J. Wright. Numerical optimization. Springer Series in Operations Research, New york, 1999. 43, 44, 45, 47, 48, 49, 51, 98, 101

[Oskiper 2007] Taragay Oskiper, Zhiwei Zhu, Supun Samarasekera et Rakesh Kumar. *Visual Odometry System Using Multiple Stereo Cameras and Inertial Measurement Unit*. Computer Vision and Pattern Recognition, 2007. 13, 136

[Peter Sturm 1999] Steve Maybank Peter Sturm. *On Plane-Based Camera Calibration : A General Algorithm, Singularities, Applications*. In Conference on Computer Vision and Pattern Recognition, 1999. 26

[Pollefeys 1997] Marc Pollefeys et Luc Van Gool. *Self-calibration from the absolute conic on the plane at infinity*. vol. 1296, pages 175–182, 1997. 26

[Pollefeys 2004] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops et R. Koch. *Visual Modeling with a Hand-Held Camera*. International Journal of Computer Vision, vol. 59, no. 3, pages 207–232, 2004. 86

[Powell 1970] M.J.D. Powell. *A hybrid method for nonlinear equations*. Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz (Ed.), pages 87–114, 1970. 49, 120

[Predmore 2010] C.Read Predmore. *Bundle adjustment of multi-position measurements using the Mahalanobis distance*. Precision Engineering, vol. 34, no. 1, pages 113 – 123, 2010. CIRP-CAT 2007. 138

[Raguram 2008] Rahul Raguram, Jan michael Frahm et Marc Pollefeys. *A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus*. In ECCV, 2008. 55

[Regiǹska 2002] Teresa Regiǹska. *Regularization of discrete ill-posed problems*. BIT Numerical Mathematics, pages 119–133, 2002. 59, 60, 140

[Ribas 2006] D. Ribas, P. Ridao, J. Neira et J.D. Tardos. *SLAM using an Imaging Sonar for Partially Structured Underwater Environments*. pages 5040 –5045, oct. 2006. 13, 136

[Royer 2006] Éric Royer. *Cartographie 3D et localisation par vision monoculaire pour la navigation autonome d'un robot mobile*. PhD thesis, Université Blaise Pascal - Clermont II, 2006. 73, 86

[Sarkis 2007] M. Sarkis, K. Diepold et K. Hüper. *A fast and robust solution to the five-point relative pose problem using gauss-newton optimization on a manifold*. In International Conference on Acoustics, Speech, and Signal Processing, 2007. 70, 72

[Saurer 2010] Olivier Saurer, Friedrich Fraundorfer et Marc Pollefeys. *OmniTour : Semi-automatic generation of interactive virtual tours from omnidirectional video*. In International Symposium on 3D Data Processing, Visualization and Transmission, 2010. 139

[Schneider 2007] D. Schneider et H.-G. Maas. *Integrated Bundle Adjustment with Variance Component Estimation : Fusion of Terrestrial Laser Scanner Data, Panoramic and Central Perspective Image Data*. In ISPRS Workshop on Laser Scanning, 2007. 140

[Servant 2010] F. Servant, P. Houlier et E. Marchand. *Improving monocular plane-based SLAM with inertial measures.* In IROS, 2010. 13, 136, 137

[Shanno 1978] D. F. Shanno. *Conjugate gradient methods with inexact searches*. In Math. Oper. Res, 1978. 47

[Shum 1999] Heung-Yeung Shum, Zhengyou Zhang et Qifa Ke. *Efficient Bundle Adjustment with Virtual Key Frames : A Hierarchical Approach to Multi-Frame Structure from Motion*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, vol. 2, page 2538, 1999. 86, 97, 99

[Sinha 2006] Sudipta N Sinha, Jan-Michael Frahm, Marc Pollefeys et Yakup Genc. *GPU-Based Video Feature Tracking and Matching*. In EDGE, 2006. 65

[Slama 1980] Chester C. Slama. Manual of photogrammetry. American Society for Photogrammetry and Remote Sensor, 4 édition, 1980. 11

[Smith 2003] P. Smith, T. Drummond et K. Roussopoulos. *Computing MAP trajectories by representing, propagating and combining PDFs over groups*. pages 1275 –1282 vol.2, oct. 2003. 139

[Snavely 2008a] N. Snavely, S. M. Seitz et R. Szeliski. *Modeling the World from Internet Photo Collections*. International Journal of Computer Vision, vol. 80, no. 2, pages 189–210, November 2008. 26

[Snavely 2008b] Noah Snavely, Steven M. Seitz et Richard Szeliski. *Skeletal sets for efficient structure from motion*. In Computer Vision and Pattern Recognition, 2008. 97, 100

[Steedly 2001] Drew Steedly et Irfan Essa. *Propagation of Innovative Information in Non-Linear Least-Squares Structure from Motion*. In ICCV, pages 223–229, 2001. 99

[Steedly 2003] D. Steedly, I. Essa et F. Dellaert. *Spectral partitioning for structure from motion*. In Computer Vision, pages 996 –1003 vol.2, oct. 2003. 99

[Strasdat 2010] H. Strasdat, J.M.M. Montiel et A.J. Davison. *Real-time monocular SLAM : Why filter ?* Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 2657 –2664, may. 2010. 139, 141

[Strecha 2010] C. Strecha, T. Pylvanainen et P. Fua. *Dynamic and Scalable Large Scale Image Reconstruction*. In Computer Vision and Pattern Recognition, 2010. 11, 97

[Strelow 2002] D. Strelow et S. Singh. *Optimal motion estimation from visual and inertial measurements*. In Workshop on Applications of Computer Vision, pages 314 – 319, 2002. 139

[Strelow 2004] Dennis Strelow. *Motion estimation from image and inertial measurements*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, November 2004. 13, 136, 137, 138

[Sturm 1996] P. Sturm et B. Triggs. *A factorization based algorithm for multi-image projective structure and motion*. In B. Buxton et Roberto Cipolla, editeurs, European Conference on Computer Vision, volume 1065 of *Lecture Notes in Computer Science*, pages 709–720. Sprin-ger-Ver-lag, April 1996. 74, 75, 76

[Sturm 1997] Peter Sturm. *Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction*. In Computer Vision and Pattern Recognition, pages 1100–1105, 1997. 26

[Tardif 2007] J-P. Tardif, A. Bartoli, M. Trudeau, N. Guilbert et S. Roy. *Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion*. In Computer Vision and Pattern Recognition, pages 1–8, 2007. 76, 120

[Titterton 2004] D. Titterton et J. Weston. Strapdown inertial navigation technology. American Institute of Aeronautics and Astronautics, 2ème édition, 2004. 30

[Torr 1997] P. H. S. Torr et D. W. Murray. *The development and comparison of robust methods for estimating the fundamental matrix*. International Journal of Computer Vision, vol. 24, pages 271–300, 1997. 70, 72, 75

[Triggs 1996] Bill Triggs. *Factorization Methods for Projective Structure and Motion*. In Computer Vision and Pattern Recognition, pages 845–851, 1996. 74

[Triggs 2000] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley et Andrew W. Fitzgibbon. *Bundle Adjustment - A Modern Synthesis*. In International Conference on Computer Vision, pages 298–372, London, UK, 2000. Springer-Verlag. 11, 77, 83, 85, 96, 97, 98, 99

[Tuytelaars 2008] Tinne Tuytelaars et Krystian Mikolajczyk. *Local invariant feature detectors : a survey*. Found. Trends. Comput. Graph. Vis., vol. 3, no. 3, pages 177–280, 2008. 64

[Wahbaa 1975] G. Wahbaa et S. Woldb. *A completely automatic french curve : fitting spline functions by cross validation*. Communications in Statistics - Theory and Methods, vol. 4, pages 1–17, 1975. 57

[Wang 2001] X. Wang et T.A. Clarke. *Separate adjustment in close-range photogrammetry*. PandRS, vol. 55, pages 289–298, 2001. 85

[Xsens 2008] Xsens. *3DOF Orientation Tracker MTx*, May 2008. `www.xsens.com/en/general/mtx`. 28

[You 2001] S. You et U. Neumann. *Fusion of vision and gyro tracking for robust augmented reality registration*. In Virtual Reality, pages 71 –78, mar. 2001. 13, 136, 167

[Yuan 2009] Gonglin Yuan, Shide Meng et Zengxin Wei. *A Trust-Region-Based BFGS Method with Line Search Technique for Symmetric Nonlinear Equations*. Advances in Operations Research, vol. 1, page 22, 2009. 47

[Zhang 1992] Z.Y. Zhang et O.D. Faugeras. 3D dynamic scene analysis : A stereo based approach. Springer, 1992. 20

[Zhang 1998] Zhengyou Zhang et T. Kanade. *Determining the Epipolar Geometry and its Uncertainty : A Review*. International Journal of Computer Vision, vol. 27, pages 161–195, 1998. 64

[Zhang 2000] Zhengyou Zhang. *A flexible new technique for camera calibration*. Pattern Analysis and Machine Intelligence, vol. 22, pages 1330 – 1334, 2000. 26

[Zhang 2001] Zhengyou Zhang et Charles Loop. *Estimating the Fundamental Matrix by Transforming Image Points in Projective Space*. Computer vision and image understanding, vol. 82, no. 2, pages 174–180, 2001. 97

[Zibetti 2008] Marcelo V. W. Zibetti, Fermín S. V. Bazán et Joceli Mayer. *Determining the regularization parameters for super-resolution problems*. In Signal Processing, volume 88, pages 2890–2901, 2008. 140

Table des matières

Remerciements **3**

Résumé et abstract **5**

Introduction **11**

1 Éléments de base **17**
1.1 Conventions de notation . . . 18
1.2 Système et environnement . . . 20
1.2.1 Modélisation de la dynamique . . . 20
1.2.2 Repères du système . . . 21
1.3 Modélisation des capteurs . . . 22
1.3.1 Caméra sténopé . . . 22
1.3.1.1 Caméra projective . . . 23
1.3.1.2 Caméra perspective . . . 24
1.3.1.3 Modélisation de la distorsion optique . . . 25
1.3.1.4 Étalonnage de caméra . . . 26
1.3.2 Capteurs de mouvement . . . 27
1.3.2.1 Gyroscopes . . . 27
1.3.2.2 Accéléromètres . . . 27
1.3.2.3 Odomètre . . . 28
1.3.2.4 Centrale inertielle (IMU) . . . 28
1.3.3 Capteurs de position . . . 29
1.3.3.1 GPS . . . 29
1.3.3.2 Système de navigation à guidage inertiel (INS) . . . 30
1.3.3.3 Laser de poursuite . . . 31
1.3.4 Récapitulatif . . . 32
1.4 Mesures de distance . . . 32
1.4.1 Distance Euclidienne . . . 33
1.4.2 Distance de Mahalanobis . . . 33
1.4.3 Distance algébrique . . . 34

2 Estimation de paramètres et sélection de modèle **35**
2.1 Formalisation du problème . . . 36
2.2 Méthodes de moindres carrés . . . 38
2.2.1 Moindres carrés linéaires . . . 38
2.2.1.1 Résolution . . . 38
2.2.1.2 Régularisation . . . 39

2.2.2 Moindres carrés non-linéaires . . . 39
2.2.2.1 Descente de gradient et gradient conjugué . . . 43
2.2.2.2 Méthode de Gauss-Newton . . . 44
2.2.2.3 Recherche linéaire . . . 45
2.2.2.4 Région de confiance . . . 49
2.2.3 Moindres carrés non-linéaires multiobjectif . . . 52
2.2.3.1 Fonction de coût à objectifs pondérés . . . 53
2.3 Estimation robuste . . . 54
2.3.1 RANSAC . . . 55
2.3.2 M-estimateurs . . . 55
2.4 Sélection de complexité . . . 56
2.4.1 Sélection par validation croisée . . . 56
2.4.2 Sélection par critère *L-curve* et variantes . . . 59
2.4.2.1 *L-Curve* . . . 59
2.4.2.2 *L-Tangent Norm* (LTN) . . . 61
2.5 Conclusion . . . 61

3 Localisation et reconstruction 3D par vision 63
3.1 Algorithmes de base . . . 64
3.1.1 Détection et mise en correspondance de points d'intérêts . . . 64
3.1.1.1 Détection de points d'intérêts . . . 64
3.1.1.2 Description de points d'intérêts . . . 64
3.1.1.3 Mise en correspondance de points d'intérêts . . . 65
3.1.2 Reconstruction d'un nuage de points 3D . . . 65
3.1.2.1 Principe . . . 65
3.1.2.2 Triangulation multivue . . . 65
3.1.2.3 Triangulation stéréo . . . 66
3.1.2.4 Triangulation optimale . . . 67
3.1.3 Calcul de pose d'une caméra . . . 67
3.1.3.1 Calcul de pose par la méthode des 3 points . . . 68
3.1.3.2 Calcul de pose robuste . . . 68
3.1.3.3 Calcul de la pose optimale . . . 70
3.1.4 Estimation du déplacement inter-image . . . 70
3.1.4.1 Géométrie épipolaire . . . 70
3.1.4.2 Calcul de déplacement inter-image par la méthode des 5 points 72
3.1.4.3 Calcul de déplacement robuste et optimal . . . 73
3.1.5 Factorisation perspective . . . 74
3.1.5.1 Formulation du problème . . . 74
3.1.5.2 Estimation des profondeurs projectives . . . 75
3.1.5.3 Résolution par décomposition en valeurs singulières . . . 76
3.1.5.4 Gestion des occultations . . . 76
3.2 Ajustement de faisceaux . . . 77

3.2.1 Énoncé du problème . . . 77
3.2.2 Erreurs de reprojection . . . 78
3.2.2.1 Erreur de reprojection géométrique . . . 79
3.2.2.2 Erreur de reprojection algébrique . . . 79
3.2.3 Résolution du problème . . . 80
3.2.3.1 Paramétrisation des déplacements . . . 81
3.2.3.2 Construction de la matrice Jacobienne . . . 81
3.2.3.3 Résolution du problème par le complément de SCHUR . . . 83
3.2.4 Ajustement par alternance . . . 85
3.3 *Structure-from-Motion* . . . 86
3.3.1 *Batch Structure-from-Motion* . . . 86
3.3.1.1 Présentation . . . 86
3.3.1.2 Rectification métrique . . . 87
3.3.2 *Structure-from-Motion* incrémental . . . 88
3.3.2.1 Présentation . . . 88
3.3.2.2 Sélection des images clés . . . 89
3.3.2.3 Reconstruction et localisation initiales . . . 90
3.3.2.4 Reconstruction et localisation incrémentales . . . 90
3.3.2.5 Ajustement de faisceaux local . . . 92
3.3.2.6 Limites de la méthode . . . 93
3.4 Conclusion . . . 93

4 Recherche linéaire pour la reconstruction 3D 95
4.1 État de l'art . . . 96
4.1.1 Problématique . . . 96
4.1.2 Méthodes existantes et limites . . . 96
4.1.2.1 Différentes méthodes d'optimisation . . . 97
4.1.2.2 Méthodes appropriées pour l'ajustement de faisceaux . . . 98
4.1.3 Positionnement . . . 100
4.2 Recherche linéaire algébrique générique . . . 101
4.2.1 Principe général de la méthode . . . 101
4.2.1.1 Une méthode de recherche linéaire exacte . . . 102
4.2.1.2 Une méthode de recherche linéaire générique . . . 103
4.2.2 Normalisation des données . . . 103
4.2.3 *G-ALS* : une recherche linéaire algébrique globale . . . 104
4.2.3.1 Présentation . . . 104
4.2.3.2 Critères de sélection et d'acceptation de l'amplitude . . . 105
4.2.4 *Two way-ALS* : une recherche linéaire algébrique bi-dimensionnelle . . . 107
4.2.4.1 Présentation . . . 107
4.2.4.2 Critères de sélection et d'acceptation des amplitudes . . . 107
4.3 Recherche linéaire algébrique pour l'ajustement de faisceaux . . . 108
4.3.1 Principe général . . . 109

4.3.2 *G-ALS* pour l'ajustement de faisceaux non calibré . . . 110
4.3.2.1 Résolution de la recherche linéaire algébrique . . . 110
4.3.2.2 Résumé de l'algorithme *G-ALS* pour l'ajustement de faisceaux 111
4.3.3 *Two way-ALS* pour l'ajustement de faisceaux non calibré . . . 111
4.3.3.1 Résolution de la recherche linéaire algébrique . . . 112
4.3.3.2 Résumé de l'algorithme *T-ALS* pour l'ajustement de faisceaux . 116
4.3.4 Ajustement de faisceaux calibré . . . 116
4.4 Résultats expérimentaux . . . 117
4.4.1 Préliminaires . . . 117
4.4.1.1 Contexte de l'évaluation . . . 118
4.4.1.2 Génération de données de synthèse . . . 118
4.4.1.3 Estimation de la solution initiale . . . 119
4.4.1.4 Méthodes de l'état de l'état de l'art comparées . . . 120
4.4.2 Résultats sur des données synthétiques . . . 121
4.4.2.1 Première expérience . . . 121
4.4.2.2 Seconde expérience : initialisation peu précise . . . 122
4.4.3 Résultats sur des données réelles . . . 126
4.4.3.1 Données réelles . . . 127
4.4.3.2 Résultats avec un algorithme de *batch Structure-from-Motion* . 128
4.4.3.3 Résultats avec un algorithme de *Structure-from-Motion* incrémental . . . 131
4.5 Discussion et conclusion . . . 133

5 Fusion de données par ajustement de faisceaux 135
5.1 État de l'art . . . 136
5.1.1 Problématique . . . 136
5.1.2 Limites des méthodes existantes . . . 137
5.1.2.1 Fusion de données par filtrage de Kalman et variantes . . . 137
5.1.2.2 Fusion de données par couplage filtre de Kalman-ajustement de faisceaux . . . 138
5.1.2.3 Fusion de données par ajustement de faisceaux . . . 138
5.1.3 Positionnement . . . 140
5.2 Fusion de données d'incertitude inconnue . . . 142
5.2.1 Contexte et exemple de contraintes . . . 142
5.2.1.1 Contexte . . . 142
5.2.1.2 Exemple de contraintes et leur application . . . 142
5.2.2 Principes de l'approche proposée . . . 144
5.2.2.1 Fonction de coût spécifique . . . 144
5.2.2.2 Ajustement de faisceaux local contraint . . . 145
5.2.2.3 Résolution séquentielle . . . 145
5.2.3 Sélection pratique des coefficients de pondération . . . 146
5.2.3.1 Principe de la méthode . . . 147

5.2.3.2 Différents critères de sélection . . . 149
5.2.4 Ajustement de faisceaux local contraint . . . 160
5.3 Application au SLAM multi-capteur . . . 161
5.3.1 Intégration dans un SLAM visuel existant . . . 161
5.3.2 Aperçu du système . . . 163
5.3.3 Recalage et mise à l'échelle métrique de la scène . . . 163
5.3.3.1 Estimation de la similitude . . . 164
5.3.4 Construction des contraintes . . . 165
5.3.4.1 Principe . . . 165
5.3.4.2 Intégration des observations . . . 166
5.3.4.3 Choix de la contrainte . . . 168
5.4 Résultats expérimentaux . . . 168
5.4.1 Préliminaires . . . 169
5.4.2 Localisation d'un véhicule . . . 170
5.4.2.1 Présentation . . . 170
5.4.2.2 Résultats . . . 171
5.4.3 Localisation en mouvement libre . . . 174
5.4.3.1 Localisation avec un système de navigation . . . 175
5.4.3.2 Localisation avec une centrale inertielle . . . 180
5.5 Discussion et conclusion . . . 183

Conclusion **185**

Bibliographie **187**

Table des matières **205**

Liste des figures **209**

Liste des tableaux **212**

Liste des algorithmes **213**

Table des figures

1 Exemples d'applications des algorithmes de *Structure-from-Motion* : (a) une reconstruction de la place Trafalgar à londres [Agarwal 2009], (b) une application de réalité augmentée [Klein 2007], (c) une reconstruction dense en ligne d'un bureau avec une caméra à faible résolution [Newcombe 2010]. 12

1.1 Aperçu du système mobile et des différents repères. 21
1.2 Le modèle géométrique d'une caméra sténopé. 23
1.3 La centrale inertielle XSENS MTi. 29
1.4 Centrale de navigation à guidage inertiel PHINS de la société IXSEA. 31
1.5 Laser de poursuite Omnitrack de la société API. 32

2.1 Recherche linéaire : estimation de la norme α du déplacement $\boldsymbol{\delta}$. Exemple sur une fonction objectif à deux variables. Deux longueurs sont proposées (α_{loc} et α^*), qui correspondent à deux solutions : $\mathbf{x}_{loc}$ est un minimum local et $\mathbf{x}^*$ est le minimum global dans la direction du déplacement $\boldsymbol{\delta}$. 46
2.2 Conditions de WOLFE. 48
2.3 Illustration de l'algorithme *Dog-Leg* et des différents déplacements mis en jeu à l'itération k. 50
2.4 Solutions possibles et frontière de Pareto pour un problème simple à deux objectifs $\{F_1, F_2\}$. 53
2.5 Exemple de score de validation croisée *leave-one-out* sur un problème de régularisation. 58
2.6 *L-Curve* : courbe du compromis entre norme de résidus et norme de la solution sur un problème de régularisation de TIKHONOV. 60

3.1 Triangulation d'un point 3D à partir de deux vues. 66
3.2 Calcul de pose à partir des correspondances 3D/2D. 69
3.3 Géométrie épipolaire. 71
3.4 Erreur de reprojection. 78
3.5 Structure de la matrice jacobienne $\mathcal{J}(\mathbf{x})$ sur un problème de SLAM monoculaire. Les éléments colorés sont constitués de valeur non nulles. 82
3.6 Structure de la matrice hessienne approximée $\mathcal{H} \approx \mathcal{B} = {\mathcal{J}_\Delta}^\top \mathcal{J}_\Delta$. 84
3.7 Schéma de fonctionnement du SLAM incrémental de MOURAGON*et al.* [Mouragnon 2006]. 89
3.8 Ajustement de faisceaux local. Les caméras et les points rouges sont optimisées, les erreurs de reprojection des caméras vertes sont intégrées dans la fonction de coût et les caméras bleues ne sont pas prises en compte. 92

4.1 Erreurs de reprojection $F(\alpha)$ (pixels) en fonction de l'amplitude α pour différentes itérations. 112

4.2 Erreur de reprojection géométrique (a) et algébrique (b) pour différentes valeurs d'amplitudes de déplacements (caméra $\alpha_{\mathbf{P}}$ et scène $\alpha_{\mathbf{Q}}$), pour 6 itérations. 115
4.3 Illustration de la scène synthétique. Les caméras sont positionnées autour de la scène et se dirigent vers le centre de celle-ci, le centre du repère. 118
4.4 Projections des points dans les images synthétiques (zones sombres). En vert il s'agit des points réels, en bleu les points bruités et en rouge les points 3D de la scène reprojetés après l'étape d'initialisation. 119
4.5 Convergences moyennes de différents algorithmes d'ajustement de faisceaux sur des ajustements calibrés (a) et non calibrés (b). L'erreur est normalisée par l'erreur RMS initiale et est une moyenne estimée sur 20 simulations. 122
4.6 Convergences moyennes de différents algorithmes de minimisation non-linéaire sur des ajustements calibrés (a) et non calibrés (b), lorsque l'initialisation des paramètres est peu précise. L'erreur est exprimée en relatif (normalisée par l'erreur RMS initiale) et est une moyenne estimée sur 20 simulations. 123
4.7 Taux d'acceptation moyens et écarts types des hypothèses, estimés sur une cinquantaine de données réelles et simulées. 125
4.8 Images issues de la séquence `Dinosaur`. 128
4.9 Images issues de la séquence `Castle`. 128
4.10 Images issues de la séquence `House`. 129
4.11 Images issues de la séquence `LesUlis` 129
4.12 Convergence des ajustement de faisceaux sur les séquences `LesUlis30` (a, 30 caméras) et `LesUlis6` (b, 6 caméras). Les courbes en haut représentent l'évolution du RMS au cours du temps et les courbes situées en bas représentent l'évolution du RMS suivant les itérations de l'optimisation. 130
4.13 Reconstruction 3D de la ville et localisation du véhicule pour la séquence `les ulis` 150 poses. 131
4.14 Évolution du RMS (pixels) par rapport au temps de calcul (en haut) et par rapport aux itérations (en bas), pour des ajustements locaux (a) et globaux (b). 132

5.1 Schéma de fonctionnement général du SLAM monoculaire contraint par un second capteur : les mesures du capteur sont intégrées dans l'ajustement de faisceaux. 141
5.2 Illustration de la fusion de la trajectoire estimée par un algorithme de *Structure-from-Motion* séquentiel avec les observations d'un second capteur k et par l'estimation dynamique des coefficients de pondération $\lambda_{i\in[1..N_m]}$. 144
5.3 Illustration de l'estimation incrémentale des coefficients de pondération λ_{i-1} et λ_i. Ces valeurs correspondent aux maxima des scores estimés pour plusieurs valeurs de λ. 147
5.4 Exemple de critère par validation croisée *leave-one-out* (a) (et zoom (b)) avec un terme de contrainte d'orientation issue d'une centrale de navigation (INS), à la septième itération. 151

5.5 Exemple de courbe de compromis en échelle logarithmique (a) (et zoom (b)) avec un terme de contrainte d'orientation issue d'une centrale de navigation, pour l'estimation du poids λ_7. 153
5.6 Exemple de critère par L-Curve (a) avec un terme de contrainte d'orientation issue d'une centrale de navigation. La figure (a) représente l'évolution des deux objectifs (erreurs de reprojection et contrainte) par rapport aux hypothèses de poids et la figure (b), l'évolution du critère. 155
5.7 Sélection du poids λ_7 par le critère LTN, avec un terme de contrainte d'orientation issue d'une centrale de navigation. 156
5.8 Évolution des coefficients de pondération sélectionnés par les 3 méthodes (de validation croisée *leave-one-out*, L-Curve et LTN), pour des contraintes de rotation inter-caméras clés ($\varepsilon_i^{k,dR}$). La ligne rouge symbolise la 7ième image clé. . . . 158
5.9 Temps de calcul de la sélection du coefficient de pondération parmi 100 valeurs, avec une contrainte d'orientation (en matrices 5.9(a) et en reprojection 5.9(b)), pour les méthodes de validation croisée *leave-one-out*, L-Curve et LTN. Les courbes rouges représentent les temps d'optimisation sans la sélection de poids. . 159
5.10 Schéma de fonctionnement du SLAM avec un second capteur de mouvement. . . 162
5.11 Illustration de l'acquisition asynchrone des observations provenant du capteur caméra c et du capteur k. 166
5.12 Images issues de la séquence `Odiaac`. 170
5.13 Localisations par SLAM obtenues sur la séquence `Odiaac`. 172
5.14 Localisations par SLAM obtenues sur la séquence `Odiaac`. 173
5.15 Moyenne des carrés des résidus 3D sur la séquence `Odiaac` avec une contrainte d'échelle. 174
5.16 Évolution des poids sélectionnés par les méthodes de validation croisée, L-Curve et LTN, avec une contrainte sur la rotation inter-caméras clés. 175
5.17 Illustration du projet Gyroviz. 176
5.18 Images issues de la séquence `Gyroviz`. 177
5.19 Localisations obtenues par les différents SLAM sur la séquence `Gyroviz`. Vue de dessus avec un léger décalage vers le bas. 178
5.20 Comparaison des méthodes de fusion des orientations sur la séquence `Gyroviz`. 179
5.21 Moyennes des erreurs de localisation globale pour différentes valeur fixes de pondérations λ_i. 180
5.22 Photographie du dispositif de localisation ici placé sur un véhicule. Ce dispositif est composé de la caméra basse résolution AVT GUPPY et de la centrale inertielle XSENS-MTx. 181
5.23 Images issues de la séquence `Couloir`. 181
5.24 Trajectoires et reconstructions de la scène pour la séquence `Couloir`. Vues de dessus avec un léger décalage vers le bas. 182

Liste des tableaux

1.1 Nomenclature des capteurs utilisés. 32

4.1 Temps et nombre d'itérations moyens réalisés par les méthodes d'optimisation pour converger sur les données de synthèse. 124
4.2 Taux d'acceptation moyens des hypothèses *G-ALS* et *T-ALS* suivant les itérations de l'optimisation, sur une cinquantaine de données réelles et simulées. 125
4.3 Nombre (et fréquence) de racines du polynôme de degré 3 (*G-ALS*) ou 5 (*Two way-ALS*), calculés sur de multiples données réelles et simulées. 126
4.4 Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur un même jeu de données de synthèse. 126
4.5 Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur le jeu de données `LesUlis30`. 127
4.6 Exemple de valeurs d'amplitudes estimées par la recherche linéaire algébrique globale *G-ALS* et à deux dimensions *Two way-ALS* sur le jeu de données `LesUlis6`. 127
4.7 Résultats des ajustements de faisceaux calibrés par la méthode LEVENBERG-MARQUARDT sur les séquences réelles, sans et avec la recherche linéaire algébrique (globale *G-ALS* et à deux dimensions *T-ALS*). 129

5.1 Exemples de contraintes sur les paramètres intrinsèques et extrinsèques de la caméra i construites à partir des observations issues d'une seconde source ou d'*a priori*. 143
5.2 Valeurs moyennes et écarts types des poids estimés par les trois méthodes de sélection (Validation croisée, L-Curve, LTN) sur la séquence `Gyroviz` 5.4.3.1. Ces valeurs sont estimées sur une contrainte d'orientation ($\varepsilon_i^{k,dR}(\mathbf{x})$) parmi 100 hypothèses allant de $\lambda^{min} = 10^{-3}$ à $\lambda^{max} = 10^{3}$. 158
5.3 Tableau des temps de calculs des méthodes de sélection (Validation croisée, L-Curve, LTN) du coefficient de pondération parmi 100 hypothèses. Chaque optimisation de la pose est effectuée sur 10 itérations. 159
5.4 Statistiques de localisation et reconstruction d'un SLAM monoculaire sur la séquence `Odiaac`. 171
5.5 Statistiques de la localisation sur la séquence `Odiaac` par rapport à la vérité terrain. 171
5.6 Statistiques sur le ratio des normes des translations (échelles) des trajectoires sur la séquence `Odiaac` par rapport à la vérité terrain. 173
5.7 Statistiques des poids sélectionnés sur la séquence `Odiaac`. 173

5.8 Statistiques de la localisation par SLAM sur la séquence Gyroviz. 176
5.9 Statistiques sur les trajectoires reconstruites sur la séquence Gyroviz. 178
5.10 Statistiques de la localisation par SLAM sur la séquence Couloir. 181

Liste des Algorithmes

1 `NonLinearLeastSquaresLineSearch` : Résolution de moindres carrés non-linéaires par recherche linéaire 42
2 `CrossValidationModelSelection` : Algorithme de sélection de modèle par validation croisée *leave-one-out* généralisée 58
3 `CriterionBasedModelSelection` : Algorithme de sélection de modèle par critère 59
4 `BatchSfm` : Algorithme de *batch Structure-from-Motion* par factorisation 87
5 `SlamSfM` : Algorithme de *Structure-from-Motion* incrémental (SLAM monoculaire). 91
6 `StepLengthSelectionGALS` : Sélection de la meilleure amplitude validant le premier critère de WOLFE. 106
7 `StepLengthSelectionTALS` : Sélection de la meilleure amplitude validant le premier critère de WOLFE pour le *Two way-ALS*. 108
8 `GlobalAlgebraicLineSearch` : Algorithme de recherche linéaire algébrique global *G-ALS*. 113
9 `TwoWayAlgebraicLineSearch` : Algorithme de recherche linéaire algébrique *Two way-ALS* 116
10 `SfMIncBiobjectifBA` : Exemple de *Structure-from-Motion* incrémental avec un ajustement de faisceaux bi-objectif 146